IBM Business Process Manager
Version 8 Edition 5

Présentation d'IBM Business Process Manager

Les manuels PDF et le centre de documentation

Les manuels PDF sont fournis par commodité, pour impression et lecture hors ligne. Pour les dernières informations en date, voir le centre de documentation en ligne.

Les manuels PDF ont dans leur ensemble le même contenu que le centre de documentation. Certains liens présents dans les manuels en PDF sont conçus pour être utilisés dans les centres de documentation et peuvent ne pas fonctionner correctement.

Cette documentation PDF est accessible dans le trimestre qui suit une édition majeure du centre de documentation telle qu'une version 7.0 ou 7.5.

La documentation de format PDF est mise à jour moins souvent que le centre de documentation, mais plus fréquemment que les Redbooks. De manière générale, un document PDF est mis à jour lorsqu'un certain nombre de modifications a été apporté.

Table des matières

Chapitre 1. Mise en route d'IBM Business Process Manager

Cette section décrit les possibilités offertes par IBM® Business Process Manager pour gérer les processus métier et explique les relations existant entre les diverses phases de la gestion des processus métier, par exemple entre la création et le déploiement des applications de processus.

L'application de processus est le conteneur fondamental des processus et de leurs composants dans IBM Business Process Manager. Les concepteurs de processus créent des applications de processus dans les environnements de création et peuvent y inclure les services, les tâches et les artefacts nécessaires pour leur exécution.

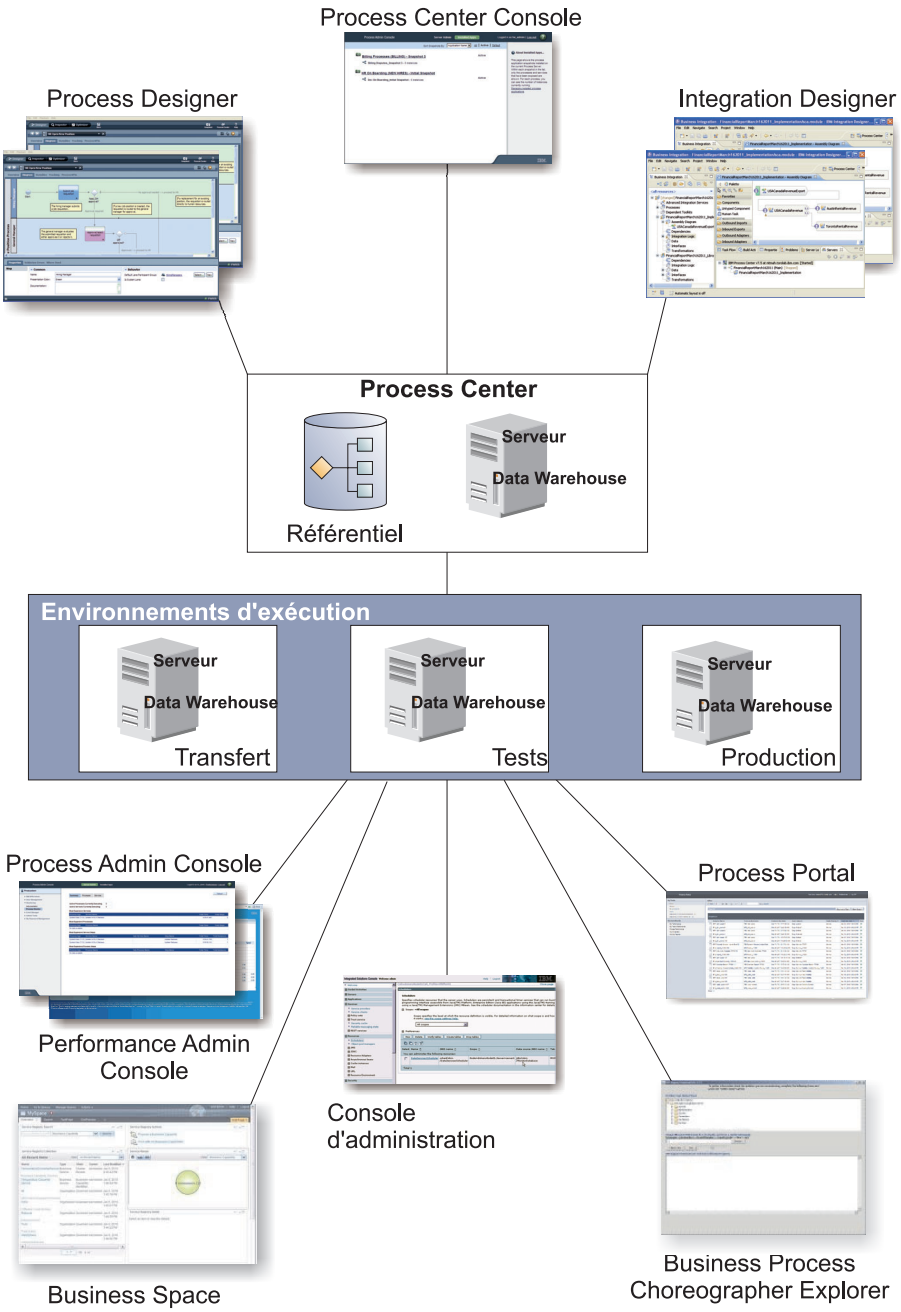
Les services d'intégration avancés sont implémentés dans IBM Integration Designer et sont associés à des applications de processus. A partir du Process Center, les applications de processus sont déployées sur le Process Server, lequel constitue l'environnement d'exécution de processus de IBM Business Process Manager.

De même, les processus automatisés créés dans Integration Designer peuvent utiliser services utilisateur développés dans IBM Process Designer.

Présentation du produit

IBM Business Process Manager est une plateforme complète de gestion des processus d'entreprise, qui offre une visibilité totale et permet de comprendre ressorts de cette gestion. Il fournit des outils et un environnement d'exécution pour la conception, l'exécution, le contrôle et l'optimisation des processus, et permet une intégration de système de base. Ce produit peut être configuré afin de prendre en charge plusieurs niveaux de complexité et d'exigence en matière de gestion de processus métier.

Les composants de IBM Business Process Manager fournissent un référentiel de gestion des processus métier unifié ; des outils pour créer, administrer et utiliser les ressources et un environnement d'exécution. Ce produit peut être configuré afin de prendre en charge plusieurs niveaux de complexité et d'exigence en matière de gestion de processus métier. Le diagramme suivant présente une configuration IBM Business Process Manager classique :



- Dans les environnements de création de IBM Process Designer et IBM Integration Designer, les développeurs se connectent à IBM Process Center. A l'aide de l'un de ces outils de développement d'application basés sur une interface graphique utilisateur, les développeurs peuvent créer, tester, déboguer et déployer des applications métier. Choisissez un outil ou l'autre, selon le type d'application que vous développez. Parfois, il existe des cas où l'utilisation de ces deux outils fournit des avantages significatifs.
- Dans les environnement de création Process Designer et Integration Designer, les concepteurs de processus et de services créent des applications de processus déployables et des kits d'outils réutilisables. Les applications de processus contiennent des modèles de processus et des implémentations de service comprenant les fichiers de prise en charge requis. Ces applications de processus sont stockées dans le référentiel Process Center afin d'être partagées.
- Process Center se compose de deux serveurs, le serveur Process Center et le serveur Performance Data Warehouse. Ils permettent aux développeurs d'Process Designer d'exécuter leurs applications de

processus et de stocker des données de performance à des fins de test et de simulation lors de la phase de développement. Performance Data Warehouse extrait des données de suivi du serveur Process Server ou Process Center à intervalles réguliers.

- Process Center prend également en charge plusieurs fonctions administratives. Dans Process Center Console, les administrateurs installent des applications de processus prêtes pour transfert, test ou mise en production sur les serveurs de processus. Les administrateurs peuvent également gérer l'exécution d'instances d'applications de processus dans des environnements configurés.
- Les applications de processus doivent être installées sur un serveur de processus à des fins de transfert, de test et de production. Les environnements de production prennent en charge les processus BPMN (Business Process Model and Notation) 2.0. IBM Business Process Manager Advanced prend également en charge les processus BPEL (Business Process Execution Language).
- A partir de Admin Console et de Performance Admin Console, les administrateurs peuvent gérer tous les serveurs d'exécution et assurer leur maintenance. Process Admin Console permet de gérer le serveur Process Center et les serveurs de processus dans vos environnements d'exécution. Performance Admin Console permet d'identifier les goulots d'étranglement de performances et de capturer des données d'instrumentation pour effectuer une analyse approfondie.
- La console d'administration permet de créer et de gérer des objets comme des ressources, des applications et des serveurs. Elle permet également de consulter les messages du produit.
- Business Space permet de créer des espaces métier personnalisés fournissant des widgets pour la surveillance ou l'administration des différents aspects de votre système. Par exemple, vous pouvez surveiller des activités métier, des services et l'état du système ou administrer des règles de médiation et des calendriers métier. Vous pouvez également créer un espace métier avec les widgets de gestion des tâches manuelles et l'utiliser pour participer à des processus métier.
- Grâce au Process Portal, les participants au processus peuvent se connecter au serveur Process Center ou à Process Server dans n'importe quel environnement d'exécution configuré, qu'un processus soit développé, testé ou publié dans un environnement de production.
- Gérez les instances de processus BPEL (Business Process Execution Language) dans l'explorateur Business Process Choreographer ou dans Business Space.

Configurations d'IBM Business Process Manager

Les différentes configurations possibles de IBM Business Process Manager correspondent aux différents points d'entrée ou stades du programme de gestion des processus métier d'une entreprise.

Tableau 1. Configurations d'IBM Business Process Manager

Configuration	Phase
Avancé	**Transformation** Ensemble complet de fonctions de gestion des processus métier • Prise en charge étendue pour l'automatisation de processus à grande échelle • Intégration de composants SOA pour l'orchestration et l'intégration des services à l'échelle de l'entreprise
Standard	**Programme** Configuré pour les projets de gestion des processus métier standard • Pour les programmes d'amélioration multi-projet, avec impact important sur l'entreprise • Prise en charge de base de l'intégration de système • Gain rapide de valeur ajoutée et amélioration de la productivité des utilisateurs

Tableau 1. Configurations d'IBM Business Process Manager (suite)

Configuration	Phase
Express	**Projet** Configuré pour le premier projet de gestion des processus métier • Gain rapide de valeur ajoutée et amélioration de la productivité des utilisateurs • Prix d'entrée de gamme • Installation et configuration faciles

Capacités de configuration d'IBM Business Process Manager

Cette section présente les capacités des produits proposés par IBM pour la gestion des processus métier et vous explique comment choisir celui qui convient à votre entreprise.

IBM Business Process Manager est une plateforme unique de gestion des processus métier qui associe des fonctions humaines et des fonctions centrées sur l'intégration dans un produit unifié. Le produit propose différentes configurations possibles pour satisfaire différents utilisateurs et différents besoins dans l'entreprise. Les configurations de produit peuvent se combiner pour permettre le développement collaboratif et autoriser les environnements d'exécution déployés en réseau.

Tableau 2. Capacités de configuration d'IBM Business Process Manager

Aptitude	Adv.	Std.	Express
Exécution compatible avec WebSphere Lombardi Edition	X	X	X
Process Designer (BPMN)	X	X	X
Edition collaborative / exécution immédiate	X	X	X
Interfaces utilisateur "coach de processus" interactives	X	X	X
Règles de processus basées sur ODM	X	X	X
Process Portal	X	X	X
Génération de rapports et surveillance en temps réel	X	X	X
Outils d'analyse & optimiseur de performances	X	X	X
Performance Data Warehouse	X	X	X
Process Center / référentiel d'actifs partagés	X	X	X
Nombre illimité d'utilisateurs finaux et de créateurs de processus	X	X	200 utilisateurs / 3 créateurs
Accessibilité avancée : mise en grappe et nombre de coeurs illimité	X	X	• Production à 4 coeurs • Développement à 2 coeurs • Pas de cluster
Exécution compatible avec WebSphere Process Server	X		
Integration Designer (BPEL / SOA)	X		
Bus de service d'entreprise intégré (ESB)	X		
Support de transaction	X		
Adaptateurs d'intégration	X		
Interface utilisateur Business Space flexible	X		
Prise en charge de plateforme avancée (Linux on System z, IBM AIX, Solaris)	X	X	*Remarque

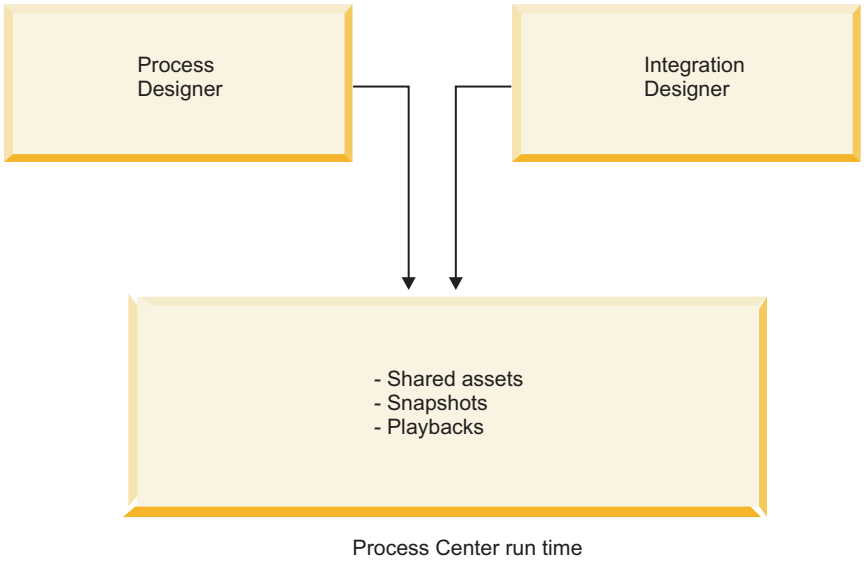
Remarque : IBM BPM Express est pris en charge sous AIX uniquement pour les clients MDM (IBM Master Data Management).

Référentiel Process Center

Process Center fournit un référentiel pour l'ensemble des processus, services et autres actifs créés dans les environnements auteur IBM Business Process Manager, Process Designer et Integration Designer.

Process Center est un composant logiciel qui s'exécute comme un serveur permettant le partages d'actifs entre Process Designer et Integration Designer, qui peuvent ainsi développer des processus métier sur un mode coopératif et hautement interactif.

Dans le diagramme suivant, plusieurs composants associés sont représentés : ceux-ci vous permettent de construire des processus métier complexes.



La console Process Center Console fournit les outils nécessaires à la maintenance du référentiel.

- A partir de la console Process Center Console, vous pouvez créer des applications de processus et des toolkits et en accorder l'accès à d'autres utilisateurs.
- Dans les environnements auteur, vous pouvez créer des modèles de processus, des services et d'autres actifs au sein d'applications de processus.
- Process Center comprend deux serveurs, un serveur Process Center et un serveur Performance Data Warehouse. Le serveur Process Center permet aux développeurs qui utilisent Process Designer d'exécuter leurs applications de processus et de stocker les données de performances à des fins de test et de simulation lors de la phase de développement. Performance Data Warehouse extrait les données suivies du serveur Process Server ou Process Center à intervalles réguliers.
- A partir de la console Process Center Console, les administrateurs installent des applications de processus prêtes pour le test ou la production sur les serveurs Process Server figurant dans ces environnements.
- A partir de la console Process Center Console, les administrateurs gèrent les instances d'applications de processus actives dans les environnements configurés.

La console Process Center Console offre un emplacement idéal pour créer et gérer des conteneurs de haut niveau tels que des applications de processus et des toolkits. Pour les administrateurs qui ne travaillent pas activement dans la vue Designer, la console Process Center Console offre une infrastructure dans laquelle les analystes et les développeurs BPM peuvent élaborer leurs processus et implémentations

sous-jacentes. Une autre tâche essentielle des administrateurs consiste à gérer l'accès au référentiel Process Center Repository en octroyant les autorisations appropriées aux utilisateurs et groupes.

Les utilisateurs titulaires des autorisations requises peuvent effectuer certaines tâches administratives directement dans Process Designer etIntegration Designer. Par exemple, un développeur disposant d'un droit d'accès en écriture à l'application de processus et souhaitant capturer l'état de tous les actifs du projet à un stade significatif du développement peut créer un instantané en travaillant dans la vue Designer.

Process Server et environnements d'exécution

Process Server fournit un unique environnement d'exécution BPM (Business Performance Management) pouvant prendre en charge une grande variété de fonctions de processus métier, d'orchestration de services et d'intégration.

Dans votre environnement de création, le serveur de processus intégré de Process Center vous permet d'exécuter des processus à mesure que vous les créez. Lorsque vous êtes prêt, vous pouvez installer et exécuter ces mêmes processus sur les serveurs de processus de vos environnements d'exécution. Le composant Business Performance Data Warehouse collecte et regroupe les données des processus exécutés sur les serveurs de processus. Vous pouvez utiliser ces données ensuite pour améliorer vos processus métier.

Le composant Process Admin Console vous permet de gérer les serveurs de processus dans vos environnements d'exécution, par exemple les serveurs utilisés pour les transferts, les tests, la production, ou le serveur de processus intégré dans le Process Center.

Environnements de création

IBM Business Process Manager Advanced propose deux environnements de création. Vous pouvez utiliser IBM Process Designer pour modéliser des processus métier comprenant des tâches manuelles. Utilisez IBM Integration Designer pour construire des services autonomes ou des services qui en appellent d'autres, par exemple des services Web, des applications d'entreprise ou des applications CICS et IMS.

- «Process Designer»
- «Integration Designer», à la page 7

Process Designer

Process Designer est disponible dans toutes les éditions du produit. IBM Business Process Manager Advanced contient également Integration Designer avec les éditeurs et les adaptateurs associés.

Un processus est une unité de logique majeure dans IBM Business Process Manager. Il s'agit du conteneur de tous les composants d'une définition de processus, notamment les services, activités et passerelles ; les événements de temporisation de message et d'exception ; les lignes de séquence, règles et variables. Lorsque vous modélisez un processus, vous créez une définition de processus métier réutilisable. Process Designer et Integration Designer peuvent créer des modèles de processus pouvant contenir des tâches manuelles.

Process Designer permet de développer des processus métier. Grâce à son interface graphique, vous pouvez créer une série d'actions pour générer un processus métier puis remodeler ce processus métier au fil du temps au gré des changements de circonstances. Si une ou plusieurs activités nécessitent un accès à de grands systèmes expéditeurs ou services fournissant des données à un processus métier, par exemple pour obtenir des informations sur les clients, vous pouvez utiliser Integration Designer. Par le biais d'une interface simple, une activité dans Process Designer peut appeler un service créé dans Integration Designer. Ce service peut utiliser des flux de médiation pour transformer, router et améliorer les données et les adaptateurs afin d'accéder à un grand nombre de systèmes expéditeurs d'une manière standardisée.

Bref, Process Designer se concentre sur le processus métier et Integration Designer met l'accent sur des services automatisés pour compléter le processus métier. Voir Mise en route d'IBM Process Designer.

Tous les projets Process Designer sont contenus dans des applications de processus. Vous stockez ces applications de processus et les artefacts associés dans le référentiel Process Center. Les applications de processus peuvent partager les actifs placés dans des kits d'outils.

IBM Business Process Manager comprend plusieurs interfaces permettant de modéliser, implémenter, simuler et vérifier des processus métier. Vous créez et gérez les applications de processus, les kits d'outils, les pistes et les instantanés dans Process Center Console. Vous pouvez créer des modèles de processus, des rapports et des services simples dans Process Designer. Vous pouvez exécuter et déboguer les processus dans le composant Inspector. Vous pouvez exécuter des simulations dans le composant Optimizer.

Vous pouvez exécuter les applications de processus développées dans Process Designer à tout moment sur le serveur Process Center ou les sauvegarder dans un instantané puis les déployer sur Process Server. Ceci vaut aussi pour les services développés dans Integration Designer et associés à des applications de processus.

Un point d'extension de l'authentificateur de connexion est défini dans Process Designer pour ouvrir un point de personnalisation et permettre à la logique de connexion du client de répondre aux exigences du serveur. Lorsque l'authentification est déclenchée, Process Designer extrait l'authentificateur et appelle la logique de connexion. Le point d'extension d'authentificateur est fourni dans le format de plug-in Eclipse. Pour utiliser le plug-in d'authentificateur, reportez-vous à la rubrique : Installation d'IBM Process Designer.

Integration Designer

Integration Designer propose des éditeurs et des rubriques d'aide pour aider les développeurs à créer des services et des processus automatisés complexes (tels que les modules SCA, médiations et processus BPEL). Il est inclus dans le package IBM Business Process Manager Advanced ou comme ensemble d'outils autonome pour d'autres usages.

IBM Integration Designer a été conçu comme un environnement de développement d'intégration complet pour les développeurs d'applications intégrées. Les applications intégrées sont complexes. Elles peuvent appeler des applications sur des systèmes d'informations d'entreprise (EIS), utiliser des processus métier dans plusieurs départements ou entreprises, et appeler des applications localement ou à distance dans différentes langues et sur différents systèmes d'exploitation. Les composants sont créés et assemblés dans d'autres applications intégrées (c'est-à-dire des applications créées à partir d'un ensemble de composants) au moyen d'éditeurs graphiques. Les éditeurs graphiques présentent une couche d'abstraction entre les composants et leurs implémentations. Avec ces outils, un développeur peut assembler une application intégrée sans connaître en détail l'implémentation sous-jacente de chaque composant.

Les outils de Integration Designer reposent sur une architecture orientée services. Les composants sont des services et une application intégrée qui implique plusieurs composants est également un service. Les services créés respectent les normes industrielles communes. Les processus BPEL, qui deviennent aussi des composants, sont également créés avec des outils graphiques qui respectent la norme industrielle BPEL (Business Process Execution Language).

Dans le paradigme de Integration Designer, les composants sont assemblés dans des modules. Des importations et des exportations sont utilisées pour partager les données entre les modules. Les artefacts placés dans une bibliothèque peuvent être partagés entre plusieurs modules.

Les modules et les bibliothèques peuvent être associés à une application de processus pour être utilisés avec Process Center et en tant que services par les processus créés dans Process Designer. Dans ce cas, ils sont également déployés avec l'application de processus.

Alternativement, les modules et les bibliothèques peuvent être déployés directement dans l'environnement de test ou sur Process Server. Vous pouvez utiliser des modules de médiation pour créer des flux de médiation que vous pourrez déployer ensuite sur WebSphere Enterprise Service Bus ou Process Server.

IBM Integration Designer permet aussi de créer des types de données et des mappes XML déployables sur un dispositif WebSphere DataPower. Vous pouvez également transférer des fichiers vers et depuis un dispositif WebSphere DataPower.

Outils d'administration

IBM Business Process Manager inclut un ensemble d'outils d'administration pour vous aider à accomplir des tâches allant de l'installation et la gestion des images instantanées à l'administration des processus et travailler avec les ressources dans votre environnement informatique.

Outils de ligne de commande

IBM Business Process Manager fournit des outils de ligne de commande, des interfaces de création de scripts et des interfaces de programmation pour gérer l'environnement d'exécution.

- Les outils de ligne de commande sont des programmes simples lancés à partir d'une ligne de commande de système d'exploitation pour effectuer des tâches spécifiques. Ils permettent, entre autres, de démarrer et d'arrêter les serveurs d'applications, de vérifier l'état d'un serveur et d'ajouter ou de supprimer des noeuds.
- Le programme de script d'administration de WebSphere (outil wsadmin) est un environnement d'interpréteur de commandes puissant, dépourvu d'interface graphique, qui permet d'exécuter des opérations d'administration dans un langage de script (Jython ou Jacl) et de soumettre des programmes en langage de script pour exécution. Il gère les mêmes tâches que la console d'administration, ainsi que la plupart des tâches de la console Process Center. L'outil wsadmin est destiné aux environnements de production et aux opérations autonomes.
- Les interfaces de programmation d'administration représentent un ensemble de classes et de méthodes Java conformes à la spécification JMX (Java Management Extensions) qui facilite l'administration des objets métier et de l'architecture SCA (Service Component Architecture). Chaque interface de programmation comprend une description de son rôle, un exemple d'utilisation de l'interface ou de la classe ainsi que des références à des descriptions de méthode.

Console Process Center

La console de Process Center offre un emplacement idéal aux utilisateurs pour créer et gérer des éléments de bibliothèque de haut niveau, tels que des applications de processus et des kits d'outils. Elle permet d'offrir une infrastructure dans laquelle les analystes et les développeurs BPM peuvent élaborer leurs processus et implémentations sous-jacentes. En outre, la console Process Center fournit des outils pour gérer le référentiel, y compris la configuration des autorisations appropriées aux utilisateurs et groupes.

Accédez à la console Process Center via un navigateur Web (par exemple, **http://host:9080/ProcessCenter**).

Process Admin Console

Process Admin Console est utilisée pour administrer les serveurs de processus dans votre environnement, y compris les utilisateurs et les images instantanées installées pour chaque serveur. En outre, elle fournit des outils qui vous aident à gérer les files d'attente et des caches.

Console Process Admin comprend le Process Inspector, un outil permettant d'afficher et de gérer les instances de processus des applications de processus qui s'exécutent sur un serveur de processus spécifique.

Accédez à Process Admin Console via un navigateur Web (par exemple, **http://host:9080/ProcessAdmin**).

Business Performance Admin Console

Business Performance Admin Console fournit des outils de gestion pour Performance Data Warehouses dans votre environnement. Cet outil permet de gérer les files d'attente du serveur et de surveiller les performances du serveur.

Accédez à Business Performance Admin Console via un navigateur Web (par exemple, **http://host:9080/PerformanceAdmin**).

Console d'administration de WebSphere Application Server

La console d'administration permet d'administrer des applications, des services, ainsi que d'autres ressources au niveau de la cellule, du noeud, du serveur ou du cluster. Vous pouvez utiliser la console avec des serveurs autonomes et des gestionnaires de déploiement gérant tous les serveurs d'une cellule dans un environnement réseau.

Si vous avez installé un profil autonome, vous disposez d'un seul noeud dans son propre domaine d'administration, appelé cellule. Utilisez la console d'administration pour gérer les applications, les bus, les serveurs et les ressources dans ce domaine d'administration. De même, si vous avez installé et configuré une cellule de déploiement réseau, vous disposez d'un noeud de gestionnaire de déploiement et d'un ou de plusieurs noeuds gérés dans la même cellule. Utilisez la console d'administration pour gérer les applications, configurer les noeuds gérés dans la cellule et contrôler ces noeuds et leurs ressources.

Accédez à cette console via un navigateur Web (par exemple, **http://host:9060/ibm/console** ou **https://host:9043/ibm/console**).

Business Process Choreographer Explorer et Business Process Archive Explorer

Selon votre rôle utilisateur, ces interfaces utilisateur vous permettent de gérer les processus BPEL et les tâches manuelles créés dans IBM Integration Designer, d'utiliser vos tâches affectées, de visualiser des processus BPEL et tâches manuelles terminés qui figurent dans la base de données d'archivage ou de supprimer des processus et des tâches de l'archivage.

Widgets Administration

Les widgets d'administration permettent de gérer et de surveiller certains composants de votre solution globale de gestion des processus métier, notamment les modules et services d'intégration avancée. Dans un espace métier, ces widgets fournissent une visibilité dans votre application et modules de service et permettent de répondre aux questions suivantes :

- Quels sont les services consommés dans un module ou exposés par un module, et quels sont les délais de réponse et la capacité de traitement sur une période définie pour ces services ?
- Quel est le statut d'un module ?
- Un module contient-il des événements ayant échoué ?
- Quelles sont les règles de médiation associées au module ?
- Quels sont les processus BPEL et tâches manuelles utilisés dans un module ?
- Un module contient-il des calendriers métier ou des règles métier ?

Utilisez un ou plusieurs widgets pour obtenir un instantané de l'état global de votre solution métier dans le système, notamment le statut de votre topologie (environnements de déploiement, clusters), les applications système (par exemple : l'échec du gestionnaire d'événement ou Business Process Choreographer), les sources de données, les moteurs de messagerie et les files d'attente de messagerie.

Business Process Rules Manager

Le gestionnaire de règles métier Business Process Rules Manager est un outil basé sur le Web, conçu pour aider les analystes métier à consulter et modifier les valeurs des règles métier. Cet outil est une option d'IBM Process Server que vous pouvez choisir d'installer au moment de la création du profil ou après l'installation du serveur.

Accessibilité dans IBM Business Process Manager

Les fonctions d'accessibilité permettent aux utilisateurs atteints de handicaps physiques, tels qu'une mobilité ou une vision réduites, d'utiliser correctement les produits issus des technologies de l'information.

IBM s'efforce de proposer des produits accessibles à tous les utilisateurs, indépendamment de leur âge ou de leurs capacités.

Pour plus d'informations sur les fonctions d'accessibilité de ce produit, voir Fonctions d'accessibilité dans IBM Business Process Manager.

Disponibilité des langues nationales dans IBM Business Process Manager

IBM Business Process Manager est disponible dans un certain nombre de langues. La liste ci-dessous décrit le niveau de prise en charge d'une langue donnée.

IBM Business Process Manager prend en charge les langues ci-dessous. La documentation n'est pas nécessairement traduite dans son intégralité.

- Chinois simplifié
- Chinois traditionnel
- Tchèque
- Anglais U.S.
- Français
- Allemand
- Hongrois
- Italien
- Japonais
- Coréen
- Polonais
- Portugais (Brésil)
- Russe
- Espagnol

IBM Business Process Manager prend partiellement en charge les langues suivantes. La documentation n'est pas nécessairement traduite dans son intégralité.

- Arabe (documentation traduite pour les widgets de tâches manuelles BPEL, les widgets de l'explorateur Business Process Choreographer, l'architecture Business Space et les widgets Business Space Monitor)
- Danois (documentation traduite pour les widgets de contrôle Business Space et l'architecture Business Space)
- Néerlandais (documentation traduite pour Process Designer, Process Center, BPD Modeler, Service Modeler, JSEditor, Process Designer, l'infrastructure Business Space et le widget Business Space Monitor)
- Finnois (documentation traduite pour les widgets de contrôle Business Space, l'infrastructure Business Space, BPD Modeler, Service Modeler, JSEditor et Process Designer)
- Grec (documentation traduite pour Process Designer, Process Center et Business Space)
- Hébreu (documentation traduite pour les tâches manuelles BPEL, l'explorateur Business Process Choreographer et l'infrastructure Business Space et les widgets Business Space Monitor)

- Norvégien (documentation traduite pour les widgets de contrôle Business Space et l'architecture Business Space)
- Portugais (Portugal) (Process Designer, Process Center, BPD Modeler, Service Modeler et JSEditor)
- Roumain (documentation traduite pour les opérations d'exécution)
- Slovaque (documentation traduite pour Business Space, BPD Modeler, Service Modeler, JSEditor et Process Designer)
- Suédois (documentation traduite pour les widgets de contrôle Business Space et l'architecture Business Space)
- Turc (documentation traduite pour Business Space)

  Remarque : Pour l'environnement local turc, vous devez définir l'entrée **case-insensitive-security-cache** dans le fichier `60Database.xml` sur **false** afin que les noms d'utilisateur et les mots de passe puissent contenir la lettre *i*. Le fichier `60Database.xml` se trouve dans le répertoire *`install_root`*`\profiles\`*`nom_profil`*`\config\cells\`*`cell_name`*`\nodes\`*`nom_noeud`*`\servers\`*`nom_serveur`*`\process-center\config\system\`.

IBM Business Process Manager fournit la prise en charge permettant aux utilisateurs d'entrer des chaînes bidirectionnelles dans l'environnement Designer Process, dans les Coaches et dans Process Portal. Il fournit des API JavaScript pour la manipulation des tests de langue bidirectionnelle.

Les coaches et Process Portal prennent en charge l'utilisation des calendriers hébreu et arabe.

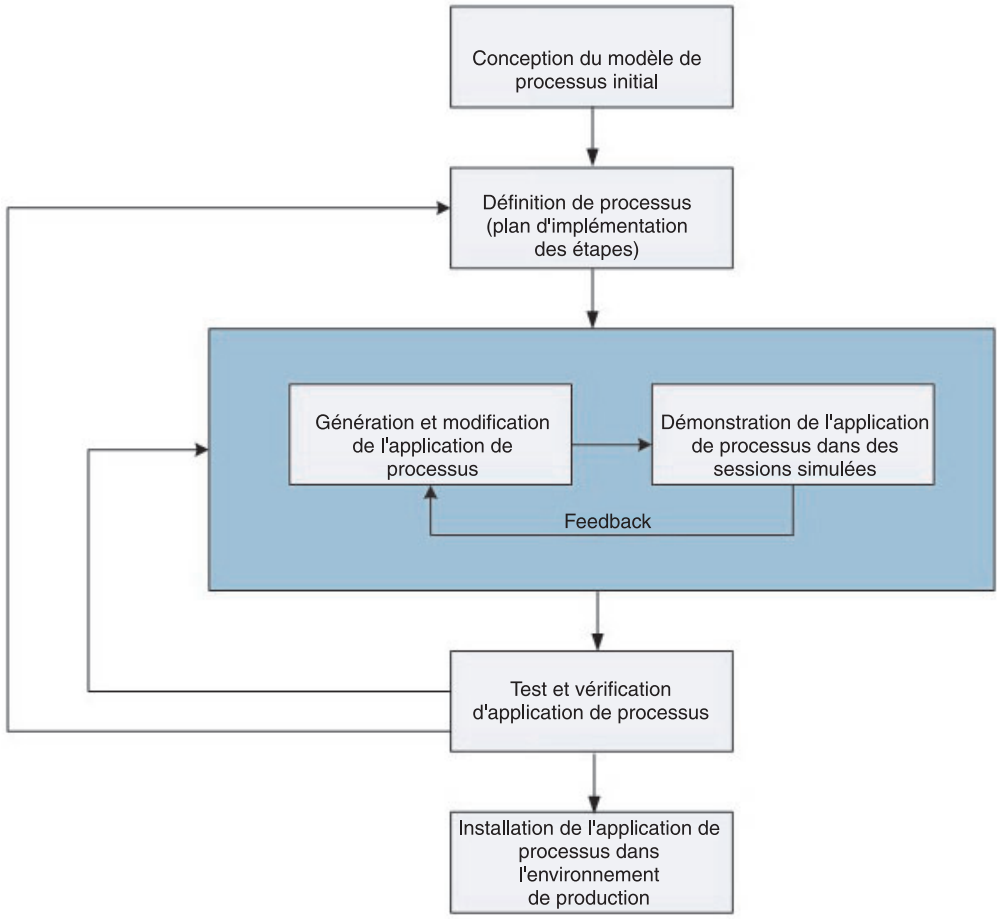
Présentation de la gestion des processus métier

Lorsque vous développez des processus dans Process Designer, vous devez prévoir l'installation éventuelle de vos applications de processus sur des serveurs dans vos environnements de test et de production.

Process Designer est disponible dans IBM Business Process Manager Express, IBM Business Process Manager Standard et IBM Business Process Manager Advanced. Dans cette section, nous allons nous intéresser à la version avancée qui est conçue pour une automatisation traitant de gros volumes et une utilisation de services complexes développés dans Integration Designer. Elle dispose de composants SOA intégrés pouvant être utilisés pour une intégration étendue des services à l'échelle de l'entreprise. La version standard peut être utilisée collégialement par de nombreux professionnels du métier afin de développer plusieurs processus sophistiqués. Elle dispose d'une intégration système de base. La version Express s'adresse à un nombre réduit d'utilisateurs dans un serveur unique et qui sont novices en matière de processus métier ou qui n'ont pas besoin d'accéder à plusieurs systèmes externes.

Le diagramme suivant illustre le cycle de vie d'un travail classique de développement de processus. Il illustre les étapes de génération et de mise au point d'un service d'installation vous permettant d'installer facilement vos applications de processus dans un environnement de production.

Comme le montre le diagramme, vous pouvez travailler exclusivement dans votre environnement de développement. Toutefois, vous devez configurer Process Server à la fois pour votre environnement de test et votre environnement de production.



Présentation de la modélisation de processus

Un processus est une unité de logique majeure dans IBM Business Process Manager. Il s'agit du conteneur de tous les composants d'une définition de processus, notamment les services, activités et passerelles ; les événements de temporisation de message et d'exception ; les lignes de séquence, règles et variables. Lorsque vous modélisez un processus, vous créez une définition de processus métier réutilisable.

Les composants de processus permettent de définir le flux de processus pour les utilisateurs finaux, par création d'une logique dans un processus et intégration à d'autres applications et sources de données. Pour comprendre ce qui se produit dans un processus au moment de l'exécution, il est essentiel de connaître les composants qui constituent un processus lors de sa conception.

Génération de processus dans IBM BPM

Un grand nombre de personnes différentes issues de diverses organisations sont généralement impliquées dans le développement de processus à l'aide d'IBM BPM. L'essentiel est de garantir que vous générez la meilleure solution possible pour atteindre les objectifs fixés pour votre projet. Pour réussir, les membres d'une équipe doivent travailler ensemble afin de cerner les exigences du processus et développer de manière itérative le modèle et ses implémentations.

Réutilisation d'éléments dans Process Designer

Process Designer permet aux développeurs de réutiliser des éléments existants dans et à travers des applications de processus. Par exemple, si vous savez qu'il existe déjà plusieurs services incluant des coaches et d'autres éléments partagés dont vous et d'autres développeurs avez besoin, vous pouvez accéder à ces éléments et les réutiliser en les incluant dans un toolkit. Ensuite, à partir de votre application de processus, vous pouvez ajouter une dépendance au toolkit contenant les éléments partagés. Cela vous permet de sélectionner l'un des services existants lorsque vous choisissez l'implémentation d'une activité. Les éléments du kit d'outils peuvent également être utilisés par d'autres développeurs qui travaillent dans d'autres applications de processus.

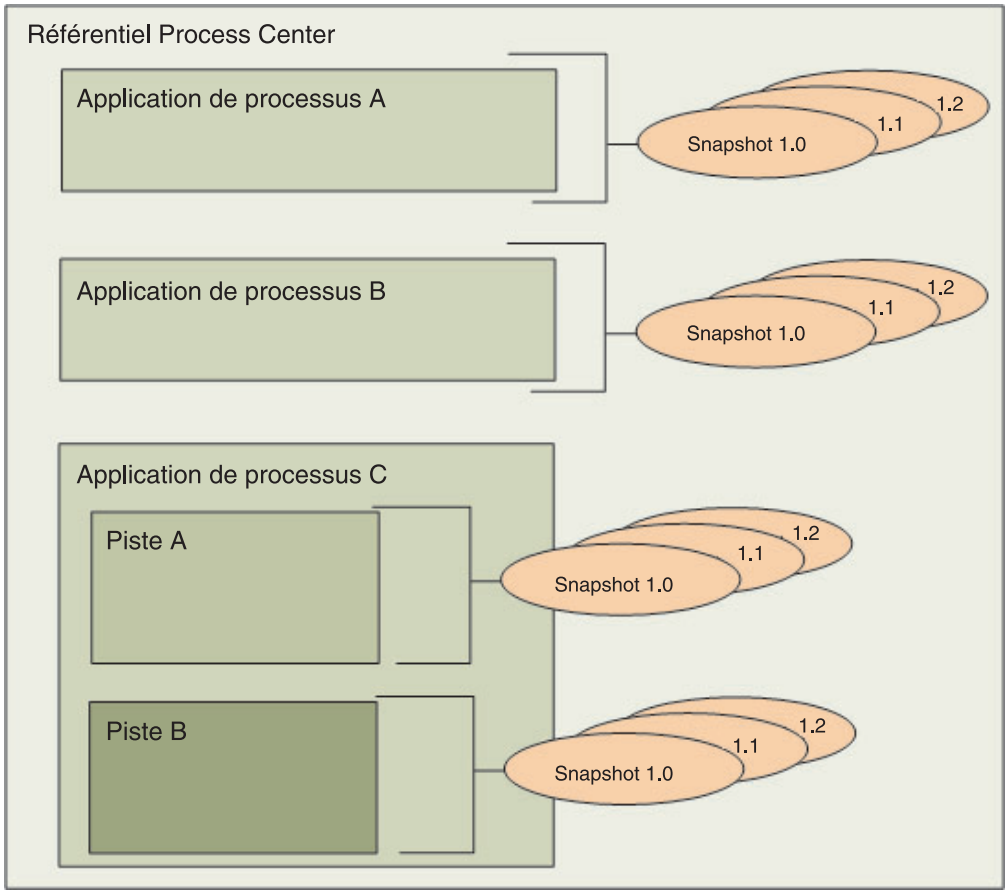
Utilisation du Designer dans IBM Process Designer

L'interface Designer fournit les outils nécessaires à la modélisation de vos processus dans IBM BPM.

Développement de processus à l'aide du Process Center

IBM Process Center sert de référentiel central à tous les actifs de projets créés dans Process Designer. Lorsque plusieurs clients Process Designer se connectent au Process Center, ils peuvent se partager des éléments, tels que des processus et des services, et visualiser les modifications apportées par d'autres utilisateurs à mesure de leur entrée. Le Process Center peut également servir de référentiel pour les actifs créés dans IBM Integration Designer.

Lorsque vous développez des processus dans Process Designer, la hiérarchie proposée dans le référentiel Process Center vous aide à gérer vos projets. La figure suivante est une représentation conceptuelle de la hiérarchie du référentiel :



Comme l'illustre le diagramme précédent, le référentiel Process Center contient les artefacts suivants :

Type de contenu	Description
Applications de processus	Conteneurs pour les modèles de processus et pour les implémentations qui les prennent en charge, que les analystes et les développeurs de modélisation des processus métier créent dans la vue Design d'IBM Process Designer.
Pistes	Subdivisions facultatives dans une application de processus déterminées par des tâches de l'équipe ou des versions de l'application de processus. Lorsqu'elles sont activées, les pistes permettent un développement parallèle. Les administrateurs déterminent si des pistes supplémentaires sont nécessaires et donc activées pour chaque application de processus.
Images instantanées	Enregistrent l'état des éléments d'une application de processus ou d'une piste à un moment déterminé. Les images instantanées représentent généralement un jalon ou servent pour les lectures ou les installations. Vous pouvez comparer des images instantanées et revenir à des images instantanées précédentes. Si un administrateur active des pistes pour une application de processus, une image instantanée est utilisée comme base pour une nouvelle piste.

Applications de processus : présentation

Une application de processus est un conteneur renfermant des modèles de processus et les implémentations qui les prennent en charge. Elle est stockée dans le référentiel. Une fois les artefacts créés par un auteur ou d'une autre manière, ils sont assemblés dans une application de processus.

Les applications de processus contiennent certains ou tous les artefacts suivants :

- Un ou plusieurs modèles de processus, aussi appelés Définitions de processus métier (BPD)
- Des références à des kits d'outils
- Les services nécessaires à l'implémentation d'activités ou à l'intégration à d'autres systèmes, y compris les services d'intégration avancée
- Une ou plusieurs pistes
- Des modules et des bibliothèques SCA (Service Component Architecture) (créés dans IBM Integration Designer)
- Tout autre élément nécessaire pour exécuter le processus

Pour visionner une vidéo présentant le développement d'une application de processus itératif et d'un kit d'outils, avec des conseils, des instantanés et des pistes, voir «Iterative Process Application and Toolkit Development», disponible sur YouTube ou sur le centre de documentation d'IBM Education Assistant. Une retranscription de la vidéo est disponible.

Tips, instantanés et pistes d'une application de processus

Toute modification apportée à une application de processus est dynamiquement enregistrée dans le référentiel Process Center du tip, qui correspond à la version actuelle de l'application de processus. Vous pouvez utiliser des sessions simulées dans le tip pour tester et gérer instantanément la version actuelle de l'application de processus.

L'application de processus reste à ce niveau de tip jusqu'à ce que vous décidiez de créer un instantané qui enregistre l'état des éléments de la bibliothèque dans une application de processus ou une piste à partir d'un point temporel spécifique. En général, vous prenez un instantané chaque fois que vous êtes prêt à tester l'intégration ou que vous voulez installer l'application de processus sur un serveur Process Center ou sur un serveur de processus à des fins de test, de transfert ou de production.

Remarque : Un tip est un instantané spécial ; il s'agit du seul type d'instantané dans lequel vous pouvez modifier son contenu, mais vous pouvez uniquement l'exécuter sur un serveur Process Center. Vous ne pouvez pas installer un tip sur un serveur de processus.

Par défaut, chaque application de processus dispose d'une piste unique, appelée `Principale`. Pour autoriser un développement parallèle sur une application de processus, vous pouvez créer des pistes

supplémentaires. Ces subdivisions facultatives d'une application de processus isole les modifications. Par exemple, imaginez que votre entreprise est en train de définir une nouvelle stratégie de marque ; au cours de cette transition, les applications de processus actuelles doivent être conservées alors que de nouvelles versions sont en cours de développement en fonction de la mise à jour de l'identité d'entreprise. Dans cette situation, une équipe peut apporter des correctifs mineurs à la version actuelle d'une application de processus (dans la piste Principale) alors qu'une autre équipe crée une nouvelle version de l'application de processus dans une piste distincte.

Kits d'outils pour les applications de processus

Les kits d'outils sont des conteneurs qui stockent des éléments de bibliothèque (par exemple : définitions de processus métier) afin d'être réutilisés par les applications de processus ou d'autres kits d'outils. Les applications de processus peuvent partager des éléments de bibliothèque issus d'un ou plusieurs kits d'outils et des kits d'outils peuvent partager des éléments de bibliothèque d'autres kits d'outils. Si vous avez accès à un kit d'outils, vous pouvez créer une dépendance par rapport à celui-ci et utiliser ses éléments de bibliothèque dans votre application de processus.

Applications de processus et applications de niveau métier

Une application de processus est associée à une application de niveau métier (BLA) qui fait office de conteneur pour l'application de processus et ses actifs (qui comprennent notamment des , des modules SCA, des kits d'outils et des bibliothèques). Chaque instantané d'application de processus dispose de son propre BLA. Un grand nombre de tâches d'administration d'un instantané (son arrêt ou son démarrage sur un serveur de production par exemple) sont effectuées au niveau de la BLA, ce qui permet d'administrer plus rapidement et plus simplement l'instantané et tous ses actifs.

Application de processus itératif et développement du kits d'outils

Tableau 3. Introduction

Scène	Audio	Action à l'écran
1	Bienvenue dans cette présentation de l'application de processus itératif et du développement de kit d'outils dans IBM Business Process Manager.	L'écran d'accueil affiche le titre de la vidéo, *Iterative process application and toolkit development*, et le sous-titre, *Learn about using tops snapshots and tracks*. Copyright 2013, IBM Corporation.
2	Dans cette vidéo, vous allez en apprendre davantage sur les application de processus et les toolkits. Vous verrez également comment vous pouvez utiliser les tips, les instantanés et les suivis pour gérer leur développement et leur déploiement.	Une liste à puces des sujets présentés dans la vidéo est affichée.
3	Les processus changent au fur et à mesure de l'évolution des besoins de votre organisation. Lorsque vous application de processus et kits d'outils passent par les phases de développement, de test et de production, vous pouvez gérer les actifs et les versions avec IBM Business Process Manager.	Un diagramme du cycle de conception itératif est fourni à titre d'illustration. Des flèches traversent le diagramme pour montrer la nature cyclique du processus de conception.
4	Dans IBM Business Process Manager, les applications de processus servent de conteneurs pour les modèles de processus et la prise en charge d'implémentations que les développeurs créent dans IBM Process Designer.	L'onglet Process Apps de la console Process Center est représenté. Le curseur survole le nom d'une application de processus puis un lien que l'utilisateur peut ouvrir dans IBM Process Designer. Le texte à l'écran «Process application: installable container for process model» s'affiche.

Tableau 3. Introduction (suite)

Scène	Audio	Action à l'écran
5	Les Toolkits sont des conteneurs qui comportent les mêmes artefacts que les applications de processus. Toutefois, à la différence des applications de processus, ils ne peuvent pas être installés et exécutés sur le serveur de processus. Les Toolkits contiennent des artefacts qui sont réutilisables par une ou plusieurs applications de processus. Les toolkits sont ensuite indirectement installés avec chaque application de processus qui leur fait référence.	L'onglet Toolkits de la console Process Center est représenté. Le curseur survole le nom d'un toolkit puis un lien que l'utilisateur peut ouvrir dans IBM Process Designer. Le texte à l'écran «Toolkit: container for reusable artifacts» s'affiche.

Tableau 4. Tips, instantanés et suivis

Scène	Audio	Action à l'écran
6	IBM Business Process Manager met en oeuvre une approche itérative pour le développement des applications de processus et des toolkits, à l'aide des fonctionnalités suivantes : • Le tip, qui est la version de travail en cours • Des instantanés, qui enregistrent l'état des éléments de bibliothèque à un moment donné • Les pistes, qui sont des branches facultatives au sein d'une application de processus ou d'un toolkit que vous pouvez utiliser pour développer plusieurs versions en même temps	Une liste à puces affiche les fonctionnalités incluses dans IBM BPM et permettant une approche itérative du développement des applications de processus et des toolkits.
7	Lors du développement d'une application de processus ou d'un toolkit dans IBM Process Designer, les modifications que vous effectuez sont sauvegardées dans le référentiel de Process Center. La version opérationnelle actuelle de votre application de processus ou toolkit est appelé "tip".	Un diagramme de processus de IBM Process Designer s'affiche. Un composant de diagramme est ouvert, et le mot *data* est remplacé par le mot *date* dans un Coach. Les modifications sont ensuite sauvegardée. Le texte à l'écran «Tip: current working version» s'affiche.
8	Vous pouvez utiliser des sessions simulées dans le tip pour tester la version actuelle de l'application de processus et du toolkit. Vous pouvez exécuter le tip uniquement sur le serveur Process Center, et vous ne pouvez installer un tip sur le serveur de processus.	La page supérieure du diagramme de processus s'ouvre et une session simulée démarre. L'interface Inspector s'ouvre et affiche le Coach modifié dans une fenêtre de navigateur. Le texte à l'écran «Tip: playback for instant testing» s'affiche.
9	Lorsque vous avez terminé de modifier votre application de processus ou toolkit, vous pouvez en créer un instantané. Les instantanés enregistrent l'état de tous les éléments d'une application de processus ou d'un toolkit à un moment déterminé. Une fois l'instantané créé, vos processus et élément associés sont toujours disponibles pour édition dans le tip. Dans Process Designer, les instantanés disponibles figurent sous Historique de révision. Même si vous pouvez afficher ces instantanés, vous ne pouvez pas les éditer.	La fenêtre de navigateur se ferme et un instantané de l'application de processus est créé. L'interface de Designer s'affiche et le curseur survole le nom du nouvel instantané sous la section Historique de révision. Le texte à l'écran «Snapshot: records the state of all items» s'affiche.

Tableau 4. Tips, instantanés et suivis (suite)

Scène	Audio	Action à l'écran
10	Vous pouvez également comparer les instantanés afin de consulter des informations telles que l'heure à laquelle ils ont été créés et les processus qui leur ont été ajoutés.	La vue de comparaison des instantanés s'affiche. Le curseur survole la date de création d'un instantané et le nom d'un élément qui a été modifié dans le processus. Le texte à l'écran «Compare changes between snapshots» s'affiche.
11	Lorsque vous créez un nouvel instantané d'un toolkit, les applications de processus qui font référence à ce toolkit continuent à utiliser l'ancien instantané. Lorsque vous ouvrez l'application de processus dans IBM Process Designer afin de l'édite, un message d'avertissement vous indique qu'une nouvelle version de de toolkit est disponible.	La console de Process Center s'affiche. La page Toolkits s'affiche et un toolkit est ouvert. Un nouvel instantané du toolkit est créé. La page Process Apps s'ouvre et une application de processus s'ouvre dans Process Designer. Le curseur survole le message d'avertissement qui s'affiche et les éléments de menu associés. Le texte à l'écran «Process applications maintain references to old toolkit snapshot» s'affiche.
12	Lorsque votre application de processus ou votre toolkit est prêt pour le déploiement, un administrateur peut installer un ou plusieurs instantanés sur le serveur de processus à partir de la console de Process Center. Les applications de processus installées peuvent uniquement faire référence aux instantanés du toolkit et non à la version actuelle d'un toolkit.	La console de Process Center s'ouvre, puis une application de processus s'affiche. Le curseur survole un lien sur un instantané sur lequel un administrateur peut cliquer pour installer l'instantané. Le texte à l'écran «Install process application snapshots on process server» s'affiche.
13	Les administrateurs peuvent également gérer les instantanés à partir de la console de Process Center, notamment créer, archiver et exporter des instantanés.	Le menu déroulant situé en regard du nom d'instantané s'ouvre et le curseur survole chaque option du menu. Le texte à l'écran «Manage snapshots using the Process Center console» s'affiche.
14	Par défaut, chaque application de processus ou tookit dispose d'une piste unique, appelée piste principale. Les administrateurs peuvent activer la création de pistes supplémentaires pour permettre un développement en parallèle. Ces subdivisions facultatives isolent les modifications.	La page de gestion de l'application de processus s'ouvre. L'option «Allow users to create tracks in this process app» est sélectionnée. Le texte à l'écran «Tracks: optional subdivisions that keep changes isolated» s'affiche.
15	Vous créez une nouvelle piste à partir d'un instantané. Cette opération copie l'instantané sélectionné dans la nouvelle piste. Les autres instantanés de la piste source ne sont pas copiés dans la nouvelle piste.	La page Snapshots de l'application de processus s'ouvre. Une nouvelle piste, nommée Track 2, est créée. Cette nouvelle piste est ouverte, puis le curseur survole le nom de l'instantané qui a été copié. Le texte à l'écran «Create a track from a snapshot» s'affiche.
16	Chaque piste comporte un tip distinct, ou version de travail actuelle. Dans une application de processus composée de plusieurs pistes, vous pouvez copier les actifs d'un instantané sur une piste dans le tip d'un autre piste. Dans ce cas, les actifs qui ont été révisés remplacent les actifs qui se trouvaient déjà sur la piste de destination, et les nouveaux actifs sont ajoutés.	Le nouvel instantané est ouvert, puis une page contenant les paramètres et les éléments de l'application s'affiche. La définition de processus s'affiche. Les actifs sont copiés dans le tip de la piste principale. S'affichent ensuite les pages contenant de nouveaux actifs, des actifs mis à jour et des actifs en conflit. Le texte à l'écran «Copy assets to other tracks» s'affiche.

Tableau 4. Tips, instantanés et suivis (suite)

Scène	Audio	Action à l'écran
17	Nous avons ici une représentation de la hiérarchie des instantanés et des pistes de l'application de processus dans le référentiel de Process Center. Dans ce diagramme, les applications de processus A et B comportent la seule piste par défaut, plusieurs instantanés et un seul tip. L'application de processus C comporte deux pistes, chacune contenant plusieurs instantané et un tip. les instantanés et le tip de chaque piste sont indépendants les uns des autres.	Un diagramme illustrant les relations entre les tips, les instantanés et les pistes, s'affiche. Ce diagramme montre les représentations des applications de processus suivantes : • Application de processus A, avec une piste, quatre instantanés et un tip • Application de processus B, avec une piste, trois instantanés et un tip • Application de processus C, avec deux pistes. La première piste comporte trois instantanés et un tip, et la deuxième piste comporte deux instantanés et un tip.

Tableau 5. Scénario d'utilisation des tips, instantanés et pistes

Scène	Audio	Action à l'écran
18	A présent que vous savez ce que sont les applications de processus et les toolkits et comment utiliser les tips, les instantanés et les pistes, voyons dans quel cas vous pourriez les utiliser. Nous allons suivre Robert, programmeur métier dans la société ABC.	Une photo d'un homme souriant s'affiche. La photo porte la légende «Robert, programmeur métier, société ABC».
19	Robert développe un processus qui modélise le mode d'embauche des nouveaux employés par sa société. Son processus comporte la piste principale par défaut. Robert apporte des modifications au processus dans le tip. Pour s'assurer que le processus s'exécute correctement, il lance une session simulée, laquelle exécute le tip sur le serveur Process Center.	Un diagramme représentant l'environnement de développement d'une application de processus s'affiche. Sous une section nommée Development, s'affiche un ensemble de trois rectangles imbriqués. Le rectangle extérieur représente le référentiel Process Center, le rectangle central représente l'application de processus et le rectangle le plus à l'intérieur représente la piste principale. A droite de la section Development figure une section nommée Testing. Cette section contient un seul rectangle qui représente le serveur Process Center. Au-dessous de la section Testing se trouve une section nommée Production. Cette section contient un seul rectangle qui représente le serveur de processus. Pour montrer que Robert apporte des modifications au tip, un triangle représentant le tip s'affiche sur la piste principale. Lorsque Robert lance une session simulée, une flèche montre le tip en train d'être copié sur le serveur Process Center.

Tableau 5. Scénario d'utilisation des tips, instantanés et pistes (suite)

Scène	Audio	Action à l'écran
20	Robert et son équipe ont fini de développer et de tester l'application de processus, et cette dernière est prête pour un déploiement en environnement de production. Pour ce faire, Robert crée un instantané de l'application de processus. Ensuite, Alice, l'administrateur, installe l'instantané sur le serveur de processus. La société ABC peut maintenant utiliser le processus de Robert pour lancer le processus d'embauche des nouveaux employés.	Pour montrer que Robert créer un instantané, une forme ovale représentant un instantané s'affiche en regard du tip sur la piste principale. Pour montrer qu'Alice installe l'instantané, une flèche désigne l'instantané en train d'être copié sur le serveur de processus.
21	Quelques temps plus tard, la société ABC annonce qu'elle va changer de nom et qu'elle s'appellera bientôt Société DEF. Au cours de cette période de transition, le processus d'embauche créé par Robert doit être maintenu le temps de développer une nouvelle version qui prenne en compte la nouvelle identité de la société. L'équipe de maintenance apporte de légers correctifs au processus d'origine sur la piste tandis que Robert et son équipe créent une version sous un nouveau nom de l'application d'embauche sur la piste DEF Rebrand.	Un instantané supplémentaire s'affiche pour montrer que l'application de processus a fait l'objet d'un développement supplémentaire. Un rectangle représentant la piste DEF Rebrand est ajouté à l'intérieur de l'application de processus. La piste DEF Rebrand comporte initialement un instantané et un tip, puis un autre instantané est ajouté pour montrer le développement supplémentaire apporté sur la piste.
22	Les deux équipes peuvent maintenant lancer des sessions simulées distinctes sur le serveur Process Center. Lorsque les applications de processus sont prêtés pour le déploiement, chaque piste peut installer un ou plusieurs instantanés sur le serveur de processus.	Des flèches indiquent le sens des tips des deux pistes qui s'exécutent en même temps sur le serveur Process Center. Ces flèches désignent ensuite les instantanés de chaque piste qui s'exécutent en même temps sur le serveur de processus.

Tableau 6. Conclusion

Scène	Audio	Action à l'écran
23	Dans cette vidéo, nous avons vu le fonctions des applications de processus et des toolkits et comment IBM Business Process Manager assure le développement du processus itérative à l'aide de tips, d'instantanés et de pistes.	Une liste à puces des sujets présentés dans la vidéo est affichée.
24	Pour plus d'informations sur IBM Business Process Manager et sur l'utilisation des tips, des instantanés et des pistes, visionnez nos autres vidéos sur YouTube ou consultez d'autres ressources officielles.	Une liste à puces répertoriant les ressources officielles suivantes s'affiche : • Canal WebSphereEducation sur YouTube • Documentation IBM Business Process Manager V8.5 • IBM developerWorks • IBM Education Assistant

Exécution et débogage des processus

Inspector permet aux développeurs d'exécuter des processus et des services sur le serveur Process Center ou sur l'exécution Process Server distante.

L'Inspector de IBM Process Designer est l'élément clé d'une approche itérative du développement de processus. Une équipe complète de développeurs peut utiliser l'Inspector pour présenter la conception et l'implémentation en cours d'un processus dans des sessions de simulation. Les sessions de simulation permettent de capturer des informations importantes auprès de différentes parties prenantes d'un

processus, tels que les gestionnaires, les utilisateurs finaux et les analystes d'entreprise. L'approche itérative du développement de processus garantit que les applications de processus atteignent les objectifs et répondent aux besoins de toutes les personnes impliquées.

L'Inspector de IBM Process Designer inclut plusieurs outils qui permettent d'effectuer des tâches telles que le suivi de vos environnements configurés :

Tâche	Description
Gestion d'instances de processus	Lorsque vous exécutez un processus, vous pouvez afficher toutes les instances exécutées auparavant ou en cours d'exécution sur les serveurs IBM Business Process Manager dans votre environnement. Vous pouvez, par exemple, stopper l'exécution des instances puis la relance. Vous pouvez également gérer des instances précédemment exécutées en filtrant ou en supprimant des enregistrements spécifiques.
Exécution pas à pas et débogage d'un processus	Pour une instance sélectionnée, affichez l'étape en cours d'exécution, puis progressez étape par étape dans le déroulement du processus pour évaluer pas à pas son exécution. Une arborescence du processus combinée à des indicateurs appelés jetons sur le diagramme de processus vous aident à déterminer à quel stade du processus vous vous trouvez. Vous bénéficiez également de l'affichage des variables utilisées à chaque étape ainsi que de leurs valeurs correspondantes (le cas échéant).

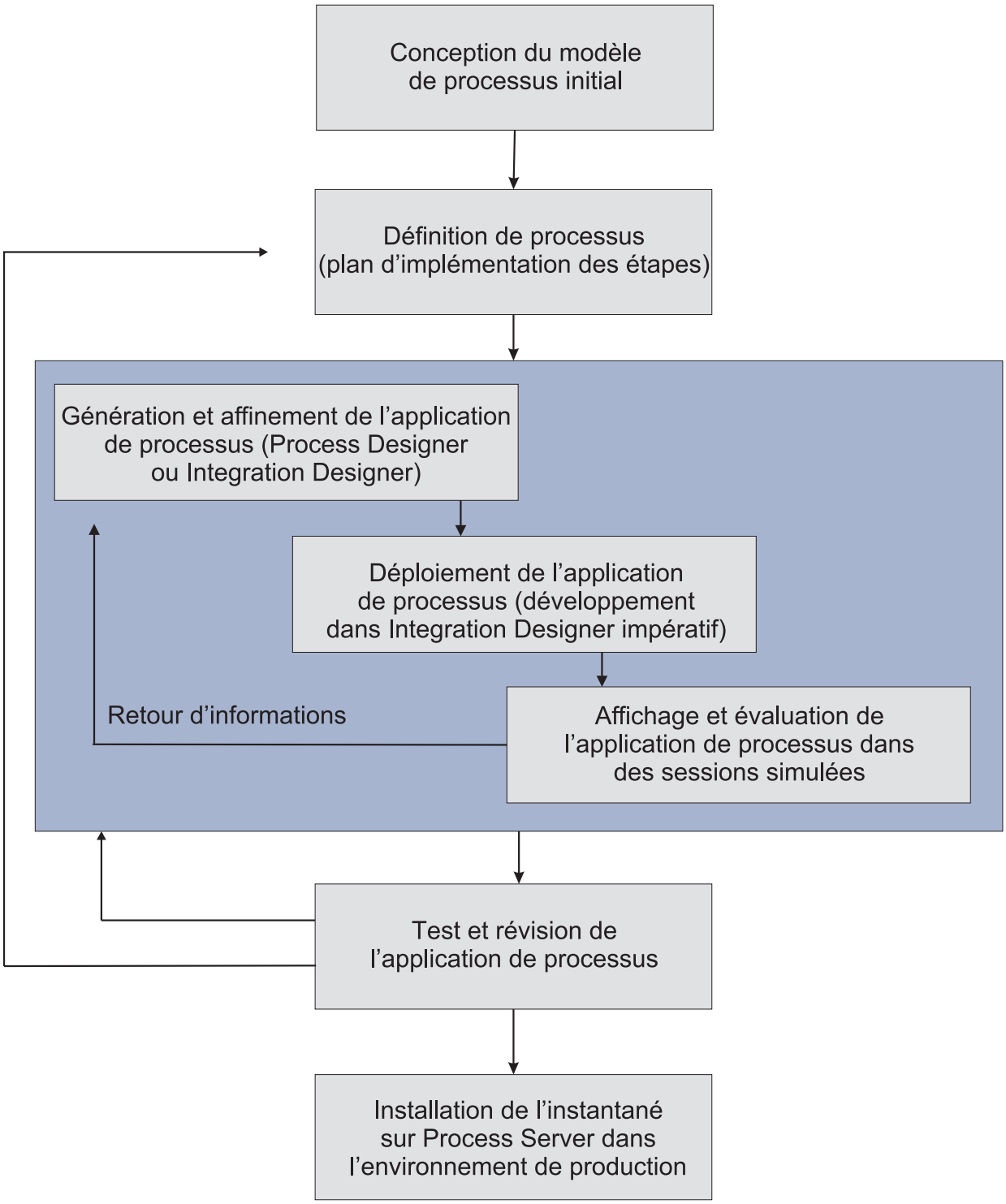
Si vous travaillez dans IBM Integration Designer, vous pouvez utiliser Inspector si votre projet est associé à une application de processus. Vous avez également d'autres outils de débogage et de test à votre disposition. Pour plus d'informations sur ces outils Integration Designer, voir "Test des modules" et "Utilisation du débogueur d'intégration pour la détermination de problème" dans les liens connexes.

Installation et gestion des applications de processus

Le cycle de vie d'une application de processus comprend l'installation, l'administration et l'annulation du déploiement des instantanés. Le cycle de vie comprend également certaines étapes liées à la gestion des versions.

Lorsque vous développez des processus, vous pouvez bénéficier pleinement de l'approche itérative prise en charge par les outils de Process Designer. Les processus évoluent avec le temps ; d'un état initial de développement, ils passent en phase de test puis en production. Mais, une fois en production, vos processus continuent de changer avec l'évolution des besoins. Se préparer aux modifications qui interviendront au cours du cycle de vie de vos processus est essentiel pour une conception efficace dès le début.

La figure suivante illustre une approche itérative du développement de processus.



Une configuration Business Process Manager classique inclut trois environnements pour prendre en charge le développement et l'installation de vos processus.

Environnement	Description
Développement	Générez et modifiez vos applications de processus dans IBM Process Designer. Créez vos modèles de processus et implémentez les étapes de ces modèles à l'aide de l'interface Designer. Avec le composant Inspector, présentez la progression de votre travail de développement dans des sessions de lecture afin d'évaluer et d'affiner rapidement votre prototype. Utilisez Process Center Console pour installer vos applications de processus sur des serveurs de processus de test ou de production.
Test	Utilisez Process Center Console pour installer vos applications de processus sur Process Server dans votre environnement de test pour implémenter des tests d'assurance qualité formalisés. L'Inspector peut vous aider à détecter et à résoudre des incidents.
Production	Une fois résolus tous les incidents détectés en phase de test, utilisez Process Center Console pour installer vos applications de processus sur le Process Server dans votre environnement de production. Vous pouvez utiliser l'Inspector pour examiner et résoudre les incidents signalés dans votre environnement de production.

Si vous souhaitez tester, installer ou administrer un instantané d'application de processus contenant des données IBM Business Process Manager Advanced, l'utilisateur ou le groupe auquel vous appartenez doit avoir reçu le rôle de sécurité d'administration Configurateur, Opérateur *et* Déployeur. Si vous ne disposez pas actuellement de ces trois rôles, cliquez sur **Utilisateurs et groupes** dans la console d'administration WebSphere pour modifier les rôles d'utilisateur ou de groupe. Voir Rôles de sécurité d'IBM Business Process Manager.

Stratégies de publication et d'installation

Pour vous assurer que les applications de processus que vous implémentez et installez respectent les normes de qualité de votre entreprise, pensez à développer une stratégie de publication et d'installation. Une fois identifiés les objectifs et les exigences de la publication et de l'installation de ces applications de processus nouvelles ou mises à jour, vous pouvez automatiser les processus nécessaires à l'approbation et au lancement des programmes.

Par exemple, vous pouvez acheminer un processus vers plusieurs responsables à travers différentes structures chargées des rapports dans votre entreprise. Le processus nouveau ou mis à jour ne pourra être installé dans votre environnement de production et transféré aux utilisateurs finaux qu'une fois validé par chaque responsable. Vous pouvez créer et implémenter dans IBM Business Process Manager Advanced les procédures de révision de ce type pour garantir que toutes les règles et obligations de l'entreprise sont respectées et que vous disposez des signatures requises. L'étape finale de la révision peut consister à envoyer une notification à l'équipe informatique indiquant que l'application de processus est prête pour l'installation.

Création, utilisation et incorporation des services

Les processus métier utilisent fréquemment des services, qui fournissent des fonctions nécessaires au processus métier. Ces services apparaissent dans un diagramme de processus métier en tant qu'activité ou étape. Par exemple, un service figurant dans Process Designer peut appeler un service Web externe ou un service complexe et automatisé conçu dans Integration Designer.

Accès à des services externes à une application

Ce scénario décrit différentes manières d'accéder à des services externes à une application et détaille les tâches de niveau supérieur à exécuter pour ce faire.

Remarque : Ce scénario s'applique à IBM Business Process Manager Advanced.

Dans une application métier intégrée, les *services métier* interagissent ensemble pour fournir une fonction demandée. Un service métier exécute une tâche ou une fonction reproductible qui participe à la réalisation d'un objectif métier donné. Toutefois, la connexion à ce service et sa recherche ne sont pas liées à la fonction métier. La séparation entre la fonction métier et la gestion des connexions au service fournit plus de flexibilité pour la solution.

L'interaction des services commence lorsqu'un *demandeur de service* adresse une demande à un *fournisseur de services* pour qu'il exécute une fonction métier. Cette demande est envoyée sous la forme d'un *message* qui définit la fonction à exécuter. Le fournisseur de services exécute la fonction demandée et envoie le résultat au demandeur de service dans un autre message. Habituellement, il faut traiter les messages pour que les services puissent échanger des données et pour implémenter d'autres fonctions informatiques de bas niveau qui sont indépendantes des données et des fonctions métier. Ceci concerne, par exemple, le routage, la conversion de protocole, la transformation, la relance d'appel en échec et l'appel de service dynamique. Ce traitement est appelé la *médiation*.

Il existe deux types de modules dans IBM Integration Designer : les modules (ou modules d'intégration métier), qui sont principalement conçus pour contenir la logique métier (processus métier, règles métier et machines d'état métier), et les modules de médiation, qui implémentent les flux de médiation. Bien que les fonctions de ces deux types de modules se recoupent parfois, il est conseillé en règle générale d'isoler la logique métier dans des modules métier et de faire exécuter la logique de médiation par des modules de médiation.

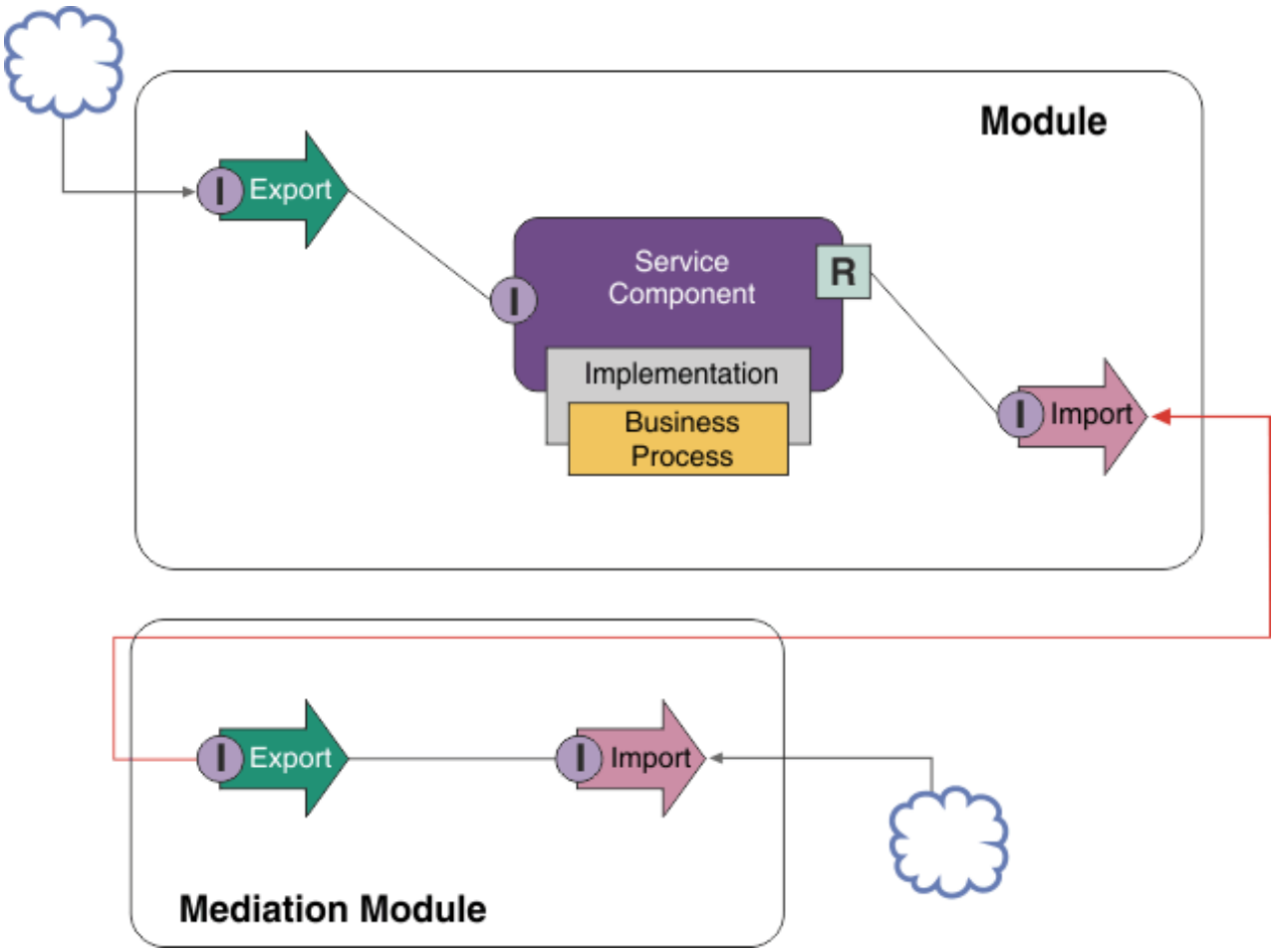
Toutefois, la séparation entre la logique métier et la logique de médiation n'est pas toujours évidente. Dans ces situations, prenez en compte les *états*, ou la quantité de données des variables qu'il faudra traiter entre les différents appels de service. En général, si le traitement des états est nul ou peu important, utilisez plutôt un composant de flux de médiation. Si vous avez besoin de stocker des données d'état entre les appels de service, ou qu'il faudra stocker des données dans des variables puis les traiter, utilisez alors un composant de processus métier. Par exemple, si vous appelez plusieurs services et que vous enregistrez les informations renvoyées par chaque appel, pour les utiliser après avoir appelé tous les services, utilisez un processus métier qui permette d'affecter facilement les données renvoyées à des variables. Autrement dit, choisissez la logique métier quand les données d'état dépassent une certaine quantité. Les sections suivantes permettent de clarifier ces conseils.

Il n'existe pas qu'un seul scénario d'intégration possible et toutes les solutions sont techniquement valides. Les recommandations détaillées ici ne sont que des pratiques conseillées pour permettre la flexibilité et la réutilisation. Comme d'habitude, évaluez soigneusement les avantages et les inconvénients de chacun de ces modèles pour votre application d'intégration métier. Examinons différentes situations.

Accès à un composant SCA

Vous accédez à un service, par exemple, quand une importation appelle un autre composant SCA sans nécessiter de transformation de données. Même dans cette situation, vous pourriez accéder au service externe à partir d'un module de médiation au lieu d'y accéder directement depuis un module métier. Cela donnerait plus de flexibilité pour la suite, pour modifier le noeud final du service, la qualité de service, ou la gouvernance (par exemple, en ajoutant la journalisation) sans conséquence pour les composants métier qui consomment le service. Ce modèle d'architecture est appelé "séparation des préoccupations".

Avant de choisir d'implémenter ce modèle, évaluez ses avantages par rapport aux effets possibles de la charge induite par un autre module. Si votre exigence principale est la flexibilité et que vous pensez changer souvent les services demandés, pensez à utiliser un module séparé, comme dans cet exemple. Si les performances comptent davantage, et que vous souhaitez mettre à jour et redéployer la logique métier, utilisez plutôt un unique module.



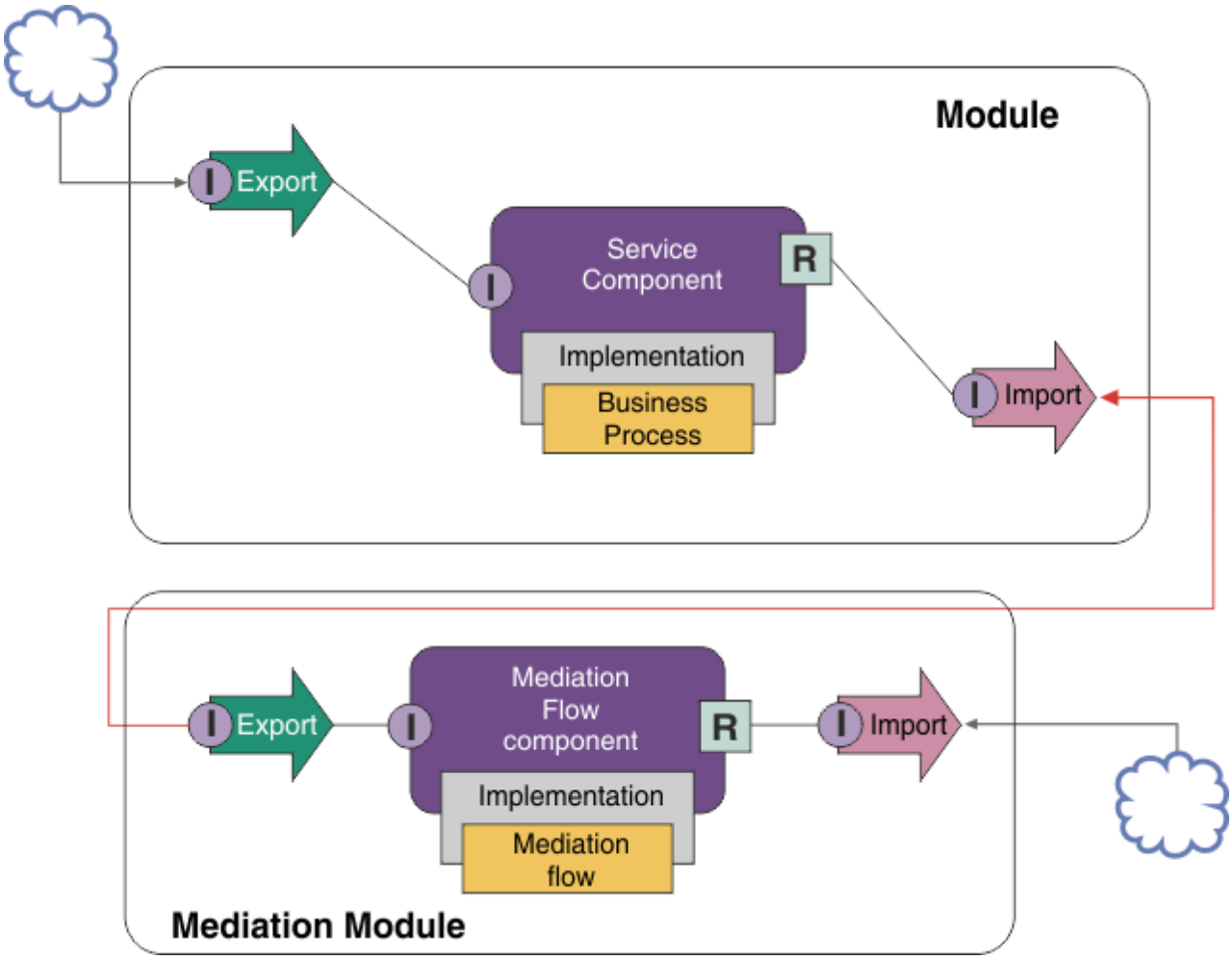
Les tâches de niveau supérieur de cet exemple sont les suivantes :

1. Créez un module de médiation. Pour obtenir les instructions associées, voir Création de modules de médiation.
2. Dans le module de médiation, créez une importation avec la liaison appropriée pour le service externe auquel vous souhaitez accéder. Pour obtenir les instructions associées, voir Création d'importations. Pour plus d'informations sur les liaisons, consultez la section Liaisons.
3. Créez une exportation et affectez-lui la même interface que pour l'importation. Pour obtenir les instructions associées, voir Création d'exportations.
4. Générez une liaison SCA pour l'exportation. Pour obtenir les instructions associées, voir Génération de liaisons SCA.
5. Dans l'assemblage du module de médiation, connectez l'exportation à l'importation. Enregistrez le module de médiation.
6. Créez un module. Pour obtenir les instructions associées, voir Création d'un module pour des services métier.
7. Ajoutez une exportation et un composant.
8. Dans la vue Intégration métier, faites glisser dans l'assemblage du module l'exportation que vous avez créée dans le module de médiation (à l'étape 4). Ceci crée une importation avec la même liaison que l'exportation.
9. Connectez l'exportation au composant puis connectez le composant à l'importation.
10. Ajoutez l'implémentation du composant. Pour plus d'informations sur les types d'implémentations, voir Implémentations.

Ensuite, vous pourrez ajouter une logique de médiation, par exemple une journalisation ou un routage, au module de médiation sans effet sur le module métier.

Ajout d'une médiation

Il ne suffit pas toujours d'appeler un service externe. Parfois, vous devez exécuter le traitement en premier, en ajoutant un module de médiation comme intermédiaire entre le demandeur de service et le fournisseur de services.



Voici quelques-unes des fonctions que le flux de médiation intermédiaire pourra effectuer :

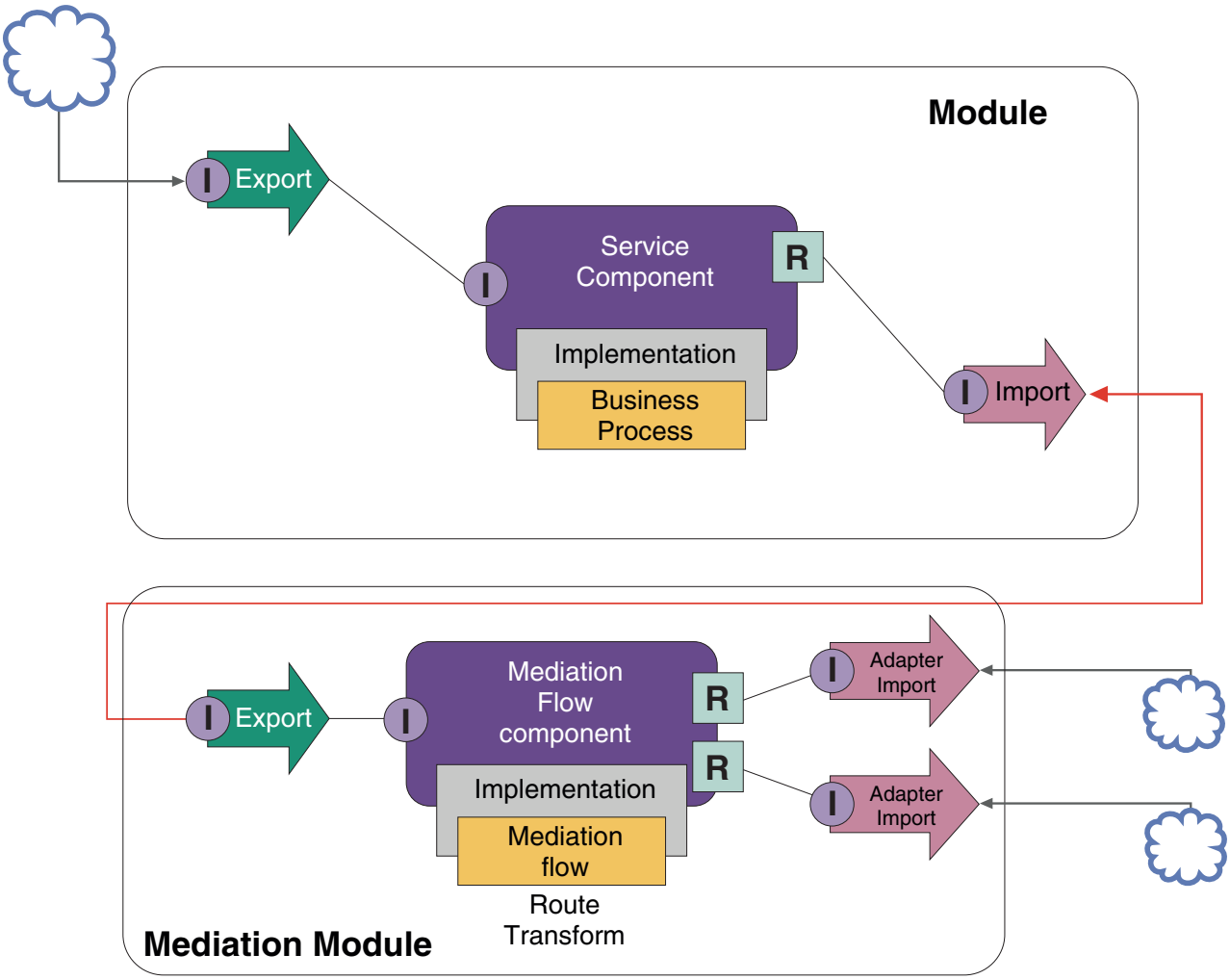
- Définir des en-têtes de protocole. Pour plus d'informations, voir la rubrique Conversion de protocole dans le centre de documentation de WebSphere Enterprise Service Bus.
- Transformer une interface ou un paramètre, avec une primitive de mappage ou une mappe d'objets métier. Voir Transformation de messages.
- Sélectionner un service particulier, avec une primitive de filtre de message. Voir Filtre de message.
- Appeler plusieurs services pour regrouper des résultats, avec les primitives de médiation de sortance et d'entrance. Voir Regroupement et diffusion de messages.
- Gérer les échecs d'appel de service en appelant le même service ou un autre service, avec une primitive Appel de service. Voir Nouvelle tentative d'appel de service
- Effectuer un routage dynamique en sélectionnant le service à utiliser en phase d'exécution, plutôt qu'en phase d'intégration, ce qui permet d'avoir des services à couplage lâche et d'anticiper les changements.

De nouveaux services peuvent être ajoutés sans toucher aux modules qui ont été déployés dans l'environnement d'exécution. Le routage dynamique est plus performant quand il utilise un registre, ce qui demande d'utiliser une primitive de médiation Recherche de noeud final. Voir Sélection dynamique des noeuds finaux.

Accès aux systèmes d'information d'entreprise

Les services et les artefacts des systèmes externes peuvent être importés dans Integration Designer. Un assistant reconnaît les applications et les données dans un système EIS (Enterprise Information System) et vous laisse générer des services à partir des applications et des données reconnues. Les artefacts générés sont des interfaces et des objets métier. Ils peuvent être utilisés par des composants dans un module.

L'utilisation d'un module de médiation intermédiaire entre un module et un système hôte facilite la réutilisation de ce système hôte. Dans l'exemple ci-dessous, on utilise un flux de médiation pour router les données vers le système hôte approprié et pour convertir les données au format requis par ce système hôte.



Les tâches de niveau supérieur de cet exemple sont les suivantes :

1. Utilisez l'assistant des services externes pour vous connecter au système hôte. L'utilisation de l'assistant des services externes pour accéder aux services externes suit le même modèle, quel que soit l'adaptateur utilisé. Pour plus d'informations sur l'utilisation de l'assistant des services externes, consultez la section Modèle d'accès aux services externes à l'aide d'adaptateurs

2. Créez un module. Pour obtenir les instructions associées, voir Création d'un module pour des services métier.
3. Ajoutez une exportation, un composant et une importation avec une liaison SCA. Pour de plus amples informations, voir Appel de services
4. Ajoutez une interface à l'exportation et connectez l'exportation au composant.
5. Ajoutez l'implémentation du composant. Dans l'implémentation, définissez une propriété indiquant le service hôte cible de l'accès. Pour plus d'informations sur les types d'implémentation, consultez la section Implémentations
6. Créez un module de médiation avec une exportation contenant une liaison SCA et la même interface que l'importation du module que vous avez créé à l'étape 2.
7. Connectez l'exportation à un composant de flux de médiation.
8. Créez une importation pour chaque système hôte auquel vous souhaitez accéder, avec l'adaptateur sortant approprié à partir de la palette de l'éditeur d'assemblage.
9. Connectez le composant de flux de médiation aux importations.
10. Implémentez le composant de flux de médiation. Utilisez une primitive Filtre de message pour choisir l'importation sur la base d'un ensemble de propriétés défini dans la logique métier et utilisez une primitive de mappage pour chaque importation d'adaptateur.
11. Dans le module, sélectionnez l'exportation du module de médiation en tant que service à importer dans le module. Pour obtenir les instructions associées, voir Appel d'un service à partir d'un autre module.

Ensuite, vous pouvez par exemple ajouter un adaptateur, ou modifier l'adaptateur pour désigner un autre système hôte, avec un effet minimal sur la logique métier.

Accès aux systèmes de messagerie

Pour que votre module SCA puisse communiquer avec un client de messagerie existant de type MQ JMS, JMS ou MQ, vous devez créer des interfaces, des objets métier et des liaisons pour les importations et les exportations. Voir Mappage d'un message à une interface SCA.

Les flux de médiation utilisent des messages qui donnent accès aux informations d'en-tête et de contexte en plus des objets métier. Si vous désirez accéder aux informations d'en-tête JMS ou à une propriété personnalisée JMS, utilisez un flux de médiation. Si vous intégrez votre application avec un système MQ et que vous souhaitez accéder aux informations d'en-tête, utilisez aussi un flux de médiation.

Création ou appel de service Web

Les services Web sont des applications intégrées exécutant des fonctions métier allant d'une simple requête à des interactions de processus métier complexes. Vous pouvez appeler un service Web existant ou développer un nouveau service Web selon vos besoins. Ce scénario décrit les étapes requises pour ce faire et fournit des informations supplémentaires.

Bien que IBM Integration Designer ne permette pas de créer tous les services dont vous avez besoin à partir de rien, certains d'entre eux seront néanmoins créés de cette manière. Lorsque vous utilisez l'éditeur d'assemblage et l'éditeur de processus métier pour assembler des services dans un processus métier, vous constatez souvent qu'il vous manque certains services. Il peut s'avérer pratique alors de créer ces services manquants avec les outils de IBM Integration Designer. L'inverse est également vrai. Après avoir créé un nouveau processus, vous pouvez trouver utile d'exposer tout ou partie des opérations de ce processus en tant que service consommable par d'autres utilisateurs.

Remarque : Ce scénario s'applique aux utilisateurs de IBM Integration Designer for IBM Process Server.

Plusieurs raisons peuvent conduire à développer des services Web avec IBM Integration Designer:

- Créer des services dans IBM Integration Designer vous permet d'implémenter ces services avec des règles métier.

- Développer dans IBM Integration Designer vous permet de développer un service Java™ et de l'exposer en tant que service Web et via SCA.
- Le mappage d'interface sans codage présente un avantage. Vous pouvez extraire tous les mappages de données depuis le code Java en laissant un programme Java boîte noire pour le développeur Java.
- IBM Integration Designer affiche tous les services et les relations au même endroit.
- La possibilité de restructurer facilite aussi le développement de services Web avec IBM Integration Designer.

N'oubliez pas que les services Web ne doivent pas être considérés comme la solution obligatoire pour tous vos problèmes d'intégration. Toutefois, comme avec toute autre technologie ou approche architecturale, il existe des avantages inhérents à l'utilisation des services Web quand vous les utilisez au bon endroit et au bon moment.

Exportations, importations et liaisons

IBM Integration Designer vous permet d'importer des services Web standard et d'utiliser ces services dans vos applications composites.

Dans IBM Integration Designer, utilisez l'éditeur d'assemblage pour développer des services. Suivez la procédure standard pour créer des modules, des modules de médiation, des bibliothèques et des composants. Ensuite, vous pouvez utiliser des exportations, des importations et des liaisons pour partager ces services et y accéder. Les étapes de ces tâches de base sont répertoriées ci-dessous et des liens conduisent à des informations plus détaillées pour chaque tâche.

Vous pouvez utiliser deux types de liaisons au choix pour les services Web : une liaison de service Web ou une liaison HTTP. Une liaison de service Web fournit une spécification permettant de transmettre des messages vers et à partir d'un service Web. Les outils permettent de générer automatiquement une liaison de service Web. Une liaison HTTP est un protocole à base de demande/réponse standard établi entre les clients et le serveur (conforme aux normes de protocole HTTP du World Wide Web Consortium (W3C). Vous devez fournir des données de configuration de liaison initiales si vous utilisez une liaison HTTP.

1. Créez une exportation pour publier le service du module afin de le rendre utilisable par les autres modules.
2. Générez une liaison pour l'exportation.
   - Générez une liaison de service Web pour l'exportation.
   - Générez une liaison d'exportation HTTP.
3. Créez une importation pour appeler un service existant ne faisant pas partie du module que vous assemblez.
   - Générez une liaison de service Web pour l'importation.
   - Générez une liaison d'importation HTTP.

Si vous souhaitez appeler un service Web à partir de pages JSP, lisez la rubrique associée au lien correspondant.

Capacités de développement de services Web

Lorsque vous ouvrez un éditeur associé au processus de création de services Web, la fenêtre de confirmation d'activation peut parfois s'ouvrir et afficher les informations suivantes :

```
This action requires the enablement of "Web Services Deployment".
Enable the required capability?
```

IBM Integration Designer fournit une fonction de filtrage appelée *Capacités*. Dans les paramètres de préférence, les fonctions et les outils sont classés par catégories de fonctionnalité et vous pouvez activer

ou désactiver ces catégories ou des ensembles de capacités dans n'importe quelle catégorie. Pour plus d'informations, voir Capacités.

En savoir plus sur les concepts clés

Utilisez cette section comme point de départ pour découvrir les technologies utilisées dans et par IBM Business Process Manager.

Scénarios de création

Utilisez des scénarios pour découvrir et utiliser les composants et produits de la famille des produits de gestion des processus métier.

Gestion des versions

Le cycle de vie d'une process application commence par la création de la process application et se poursuit au cours d'un cycle de mise à jour, de déploiement, de co-déploiement, d'annulation du déploiement et d'archivage de la process application. *La gestion des versions* est un mécanisme utilisé pour gérer le cycle de vie de l'application de processus en identifiant de façon unique les versions individuelles de l'application de processus.

La manière dont la gestion de versions fonctionne dans IBM Business Process Manager varie selon ce que vous déployez : une application de processus déployée depuis le référentiel dans IBM Process Center, ou une application d'entreprise déployée directement depuis IBM Integration Designer.

Les applications de processus et les kits d'outils que vous déployez dans un environnement d'exécution à partir de Process Center sont, par défaut, versionnés. Pour les applications d'entreprise, vous pouvez choisir de versionner les modules et les bibliothèques dans IBM Integration Designer.

De plus, vous pouvez créer des versions d'une tâche manuelle ou d'une machine d'état, de sorte que plusieurs versions de la tâche ou de la machine d'état peuvent coexister dans l'environnement d'exécution.

Gestion des versions dans les applications de processus

La gestion de versions permet à l'environnement d'exécution d'identifier les instantanés dans le cycle de vie d'une application de processus et d'exécuter simultanément plusieurs instantanés à la fois sur un serveur de processus.

Pour comprendre la manière dont les versions des applications de processus sont gérées, il est important de rappeler qu'une application de processus est un conteneur composé de plusieurs artefacts utilisés dans ou par l'application de processus (par exemple des modèles de processus ou des BPD, des références de kits d'outils, des services ou des pistes). La gestion des versions s'effectue à ce niveau du conteneur, pas au niveau des artefacts individuels. Pour les applications de processus, cela signifie que la gestion des versions s'effectue lors de la prise d'un instantané.

Vous pouvez comparer des instantanés pour identifier des différences entre les versions. Par exemple, si un développeur a résolu un problème concernant un service et pris un instantané de l'application de processus ou du kit d'outils concerné à ce stade, puis qu'un autre développeur a apporté plusieurs modifications au même service et pris un nouvel instantané, le chef de projet peut comparer ces deux instantanés pour déterminer quelles modifications ont été apportées, quand et par qui. Si le chef de projet décide que les modifications supplémentaires apportées au service ne présentent pas d'intérêt, il peut revenir à l'instantané de la rectification d'origine.

Vous pouvez exécuter simultanément des versions (instantanés) différentes d'une application de processus sur un serveur ; si vous installez un nouvel instantané, supprimez l'original ou laissez son exécution se poursuivre.

Contexte de version

Chaque instantané dispose de métadonnées uniques permettant d'identifier la version (aussi appelée "contexte de version"). Vous affectez cet identificateur, mais IBM recommande d'utiliser un système de version numérique à trois chiffres dans le format <majeur>.<mineur>.<maintenance>. Voir les rubriques sur les conventions de dénomination pour une description plus détaillées de ce schéma de gestion des versions.

IBM Business Process Manager affecte un espace de nom global pour chaque application de processus. L'espace de nom global correspond soit à l'aide de l'application de processus, soit à un instantané spécifique de l'application de processus. Le nom de version utilisé par le serveur ne doit pas dépasser sept caractères, le nom assigné est donc un acronyme qui utilise des caractères issus du nom de l'instantané que vous avez affecté. Les acronymes des instantanés sont identiques aux noms d'instantanés si ces derniers respectent le style IBM VRM recommandé et qu'ils ne dépassent pas sept caractères. Par exemple, le nom d'instantané 1.0.0 a l'acronyme 1.0.0, le nom d'instantané 10.3.0 l'acronyme 10.3.0. L'acronyme d'instantané est garanti unique dans le contexte de l'application de processus, dans la portée du serveur Process Center. C'est pourquoi vous ne pouvez pas modifier l'acronyme d'instantané.

Remarques relatives à la gestion des versions pour les applications de processus sur plusieurs clusters

Vous pouvez installer la même version d'une application de processus sur plusieurs clusters dans une même cellule. Pour faire une différence entre plusieurs installations d'une même version de l'application de processus, créez un instantané de chaque installation et incluez un ID unique de cellule dans le nom de l'instantané (par exemple : v1.0_cell1_1 et v1.0_cell1_2). Chaque instantané est une nouvelle version de l'application de processus (du point de vue de la gestion pure du cycle de vie), mais son contenu et ses fonctions sont identiques.

Lorsque vous installez une application de processus dans un cluster, une synchronisation automatique des noeuds se produit.

Remarques relatives à la gestion des versions pour les kits d'outils Process Designer

N'oubliez pas que les instantanés d'application de processus sont généralement pris lorsque vous êtes prêt à effectuer un test ou une installation. Toutefois, les instantanés d'un kits d'outils sont généralement pris lorsque vous êtes prêt à ce que ce kit d'outils soit utilisé par les applications de processus. Ensuite, pour mettre à jour le toolkit, vous devez prendre un autre instantané de la version actuelle ; les propriétaires des applications de processus et des toolkits peuvent alors décider s'ils veulent passer au nouvel instantané.

Gestion de versions de modules et de bibliothèques

Si un module ou une bibliothèque est contenu dans une application de processus ou un kit d'outils, il adopte le cycle de vie de l'application de processus ou du kit d'outils (versions, instantanés, pistes, etc.). Les noms de bibliothèque et de module doivent être uniques dans toute l'application de processus ou le kit d'outils.

Cette rubrique décrit la gestion de versions des modules et des bibliothèques qui sont utilisés avec les applications de processus. Notez que, si vous déployez des modules directement depuis IBM Integration Designer sur un Process Server, vous pouvez toujours affecter des numéros de version aux modules au cours du déploiement, comme indiqué dans la rubrique «Création de modules et de bibliothèques versionnés».

Un module ou une bibliothèque associé au IBM Process Center doit trouver ses bibliothèques dépendantes dans la même application de processus ou dans un kit d'outils dépendant.

Le tableau qui suit répertorie les sélections que vous pouvez exécuter dans l'éditeur de dépendances de IBM Integration Designer quand une bibliothèque est associée à une application de processus ou à un kit d'outils.

Tableau 7. Dépendances pour les bibliothèques Module, Application de processus ou kit d'outils et Global

Portée de la bibliothèque	**Description**	**Peut dépendre de ... . .**
Module	Il existe une copie de cette bibliothèque sur le serveur pour chaque module qui l'utilise.	Une bibliothèque ayant pour portée un module peut dépendre de tous les types de bibliothèques.
Application de processus ou kit d'outils	La bibliothèque est partagée entre tous les modules compris dans l'application de processus ou le kit d'outils. Ce paramètre prend effet si le déploiement est réalisé via le IBM Process Center. Si le déploiement a lieu en dehors du IBM Process Center, la bibliothèque est copiée dans chaque module. **Remarque :** Les bibliothèques créées dans IBM Integration Designer version 8 utilisent le niveau de partage **Application de processus ou kit d'outils** par défaut.	Une bibliothèque de ce type peut uniquement dépendre de bibliothèques globales.
Global	La bibliothèque est partagée entre tous les modules en cours d'exécution.	Une bibliothèque globale ne peut dépendre que d'autres bibliothèques globales. **Remarque :** Vous devez configurer une bibliothèque partagée WebSphere pour déployer la bibliothèque globale. Voir «Dépendances de module et de bibliothèque» pour plus d'informations.

Modules et des bibliothèques associés aux applications de processus ou kits d'outils

Il est inutile de versionner les modules et les bibliothèques associés aux applications de processus ou aux kit d'outils.

Les modules et les bibliothèques associés à une application de processus ou un kit d'outils n'ont pas besoin d'être versionnés. En fait, vous ne pouvez pas créer une version d'un module ou d'une bibliothèque avec une application de processus ou un kit d'outils dans l'éditeur de dépendance. Les modules et les bibliothèques associés à une application de processus ou un kit d'outils utilisent des images instantanées, une fonction de Process Center, pour obtenir le même résultat qu'une version.

Les bibliothèques associées à une application de processus ou un kit d'outils n'ont pas de numéro de version nécessaires dans la section Bibliothèques de l'éditeur de dépendance, car aucune version n'est nécessaire.

Conventions de dénomination

Une convention de dénomination permet de différencier les différentes versions d'une application de processus au cours des étapes de son cycle de vie que sont la mise à jour, le déploiement, le codéploiement, l'annulation de déploiement et l'archivage.

La présente section fournit les conventions qui permettent l'identification unique des versions d'une application de processus.

Un *contexte de version* est une combinaison d'acronymes décrivant de façon unique une application de processus ou un kit d'outils. Chaque type d'acronyme est doté d'une convention de dénomination. Le signe est limité à un maximum de sept caractères parmi le jeu de caractères [A à Z et 0 à 9_], hormis pour le sigle de l'instantané qui peut comprendre un point en plus.

- Le sigle de l'application de processus est créé lors de la création de l'application de processus. Il peut comporter au maximum sept caractères.
- L'acronyme de l'instantané est créé automatiquement lors de la création de cet instantané. Il peut comporter au maximum sept caractères.

  Si le nom de l'instantané répond aux critères de validité applicables aux sigles d'instantané, le nom et le sigle de l'instantané seront identiques.

  Remarque : Lors de l'utilisation de la fonction de routage adaptée à la version du composant de flux de médiation, attribuez à votre instantané un nom dans le format <*version*>.<*édition*>.<*modification*> (par exemple, **1.0.0**). Dans la mesure où le sigle de l'instantané est limité à sept caractères, les valeurs numériques sont limitées à un maximum de cinq chiffres au total (cinq chiffres plus deux points). Soyez donc attentif lors de l'incrémentation des zones numériques, dans la mesure où tout ce qui dépasse les sept premiers caractères est tronqué.

  Par exemple, le nom d'instantané **11.22.33** donne le sigle d'instantané **11.22.3**.
- Le sigle de la piste est automatiquement généré à partir du premier caractère de chaque mot composant le nom de piste. Par exemple, une nouvelle piste créée sous le nom **My New Track** donne la valeur de sigle **MNT**.

  La valeur par défaut du nom et du sigle de la piste est **Main**. Le déploiement sur un serveur IBM Process Center inclut le sigle de piste dans le contexte de gestion de versions si ce sigle de piste est différent de **Main**.

Une définition de processus métier dans une application de processus est généralement identifiée par le sigle du nom de l'application de processus, le sigle de l'instantané et le nom de la définition de processus métier. Choisissez des noms uniques pour vos définitions de processus métier chaque fois que cela s'avère possible. S'il existe des noms en double, vous pouvez rencontrer les problèmes suivants :
- Vous ne pourrez peut-être pas exposer les définitions de processus métier en tant que services Web sans une certaine forme de médiation.
- Vous ne pourrez peut-être pas appeler une définition de processus métier créée dans IBM Process Designer à partir d'un processus BPEL créé dans IBM Integration Designer.

Le contexte de version varie selon la façon dont l'application de processus est déployée.

Conventions de dénomination pour les déploiements de serveur Process Center :

Sur le serveur IBM Process Center, vous pouvez déployer un instantané d'une application de processus ainsi qu'un instantané d'un kit d'outils. De plus, vous pouvez déployer le tip d'une application de processus ou celui d'un kit d'outils. (Le terme *tip* désigne ici la version opérationelle actuelle d'une application de processus ou d'un kit d'outils). Le contexte de version varie selon le type de déploiement.

Pour les applications de processus, le tip de l'application de processus ou l'instantané d'application de processus sert à identifier de manière unique la version.

Les kits d'outils peuvent être déployés avec une ou plusieurs applications de processus mais le cycle de vie de chaque kit d'outils est lié au cycle de vie de l'application de processus. Chaque application de processus possède sa propre copie du ou des kits d'outils dépendants qui est déployée sur le serveur. Un kit d'outils déployé n'est pas partagé entre les applications de processus.

Si la piste associée à l'application de processus a un autre nom que la valeur par défaut **Main**, le sigle de la piste fait également partie du contexte de la version.

Pour plus d'informations, voir la section «Exemples», à la page 33 plus loin dans cette rubrique.

Instantanés d'application de processus

Pour les déploiements d'instantané d'application de processus, le contexte de la version combine les éléments suivants :

- Sigle du nom de l'application de processus
- Sigle de la piste de l'application de processus (si la piste utilisée n'est pas **Main**)
- Sigle de l'instantané d'application de processus

Kits d'outils autonomes

Pour les déploiements d'instantané de kit d'outils, le contexte de la version combine les éléments suivants :

- Sigle du nom du kit d'outils
- Sigle de la piste du kit d'outils (si la piste utilisée n'est pas **Main**)
- Sigle de l'instantané du kit d'outils

Tips

Les tips d'application de processus sont utilisés au cours des tests itératifs dans Process Designer. Ils peuvent être déployés uniquement sur des serveurs Process Center.

Pour les déploiements de tip d'application de processus, le contexte de la version combine les éléments suivants :

- Sigle du nom de l'application de processus
- Sigle de la piste de l'application de processus (si la piste utilisée n'est pas **Main**)
- "Tip"

Les tips de kit d'outils sont utilisés au cours des tests itératifs dans Toolkit Designer. Ils ne sont pas déployés sur un serveur de production.

Pour les déploiements de tip de kit d'outils, le contexte de la version combine les éléments suivants :

- Sigle du nom du kit d'outils
- Sigle de la piste du kit d'outils (si la piste utilisée n'est pas **Main**)
- "Tip"

Exemples

Les ressources doivent être nommées de manière unique et identifiées de façon externe avec le contexte de la version.

- Le tableau suivant contient un exemple de noms identifiés de manière unique. Dans cet exemple, un tip d'application de processus utilise le nom de piste par défaut (**Main**) :

Tableau 8. Tip d'application de processus avec le nom de piste par défaut

Type de nom	**Exemple**
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Main**
Sigle de la piste d'application de processus	"" (quand la piste est **Main**)
Instantané d'application de processus	
Sigle de l'instantané d'application de processus	**Tip**

Tous les modules SCA associés à ce tip d'application de processus contiennent le contexte de version, comme dans le tableau suivant :

Tableau 9. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-Tip-M1	PA1-Tip-M1.ear
M2	PA1-Tip-M2	PA1-Tip-M2.ear

- Le tableau suivant contient un exemple de tip d'application de processus qui utilise un nom de piste autre que le nom par défaut :

Tableau 10. Tip d'application de processus sans le nom de piste par défaut

Type de nom	Exemple
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Track1**
Sigle de la piste d'application de processus	**T1**
Instantané d'application de processus	
Sigle de l'instantané d'application de processus	**Tip**

Tous les modules SCA associés à ce tip d'application de processus contiennent le contexte de version, comme dans le tableau suivant :

Tableau 11. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-T1-Tip-M1	PA1-T1-Tip-M1.ear
M2	PA1-T1-Tip-M2	PA1-T1-Tip-M2.ear

Des conventions de dénomination similaires s'appliquent aux déploiements avancés d'instantanés et d'astuces du kit d'outils. Elles s'appliquent également aux instantanés avancés installés sur le serveur de processus.

- Le tableau suivant contient un exemple de noms identifiés de manière unique. Dans cet exemple, un instantané d'application de processus utilise le nom de piste par défaut (**Main**):

Tableau 12. Instantané d'application de processus avec un nom de piste par défaut

Type de nom	Exemple
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Main**
Sigle de la piste d'application de processus	"" (quand la piste est **Main**)
Instantané d'application de processus	**Process Shapshot V1**
Sigle de l'instantané d'application de processus	**PSV1**

Les modules SCA associés à cet instantané d'application de processus incluent le contexte de version, comme indiqué dans le tableau suivant :

Tableau 13. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-PSV1-M1	PA1-PSV1-M1.ear

Tableau 13. Modules SCA et fichiers EAR avec version (suite)

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M2	PA1-PSV1-M2	PA1-PSV1-M2.ear

- Le tableau suivant montre un exemple d'instantané d'application de processus n'utilisant pas un nom de piste par défaut :

Tableau 14. Instantané d'application de processus sans nom de piste par défaut

Type de nom	Exemple
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Track1**
Sigle de la piste d'application de processus	**T1**
Instantané d'application de processus	**Process Snapshot V1**
Sigle de l'instantané d'application de processus	**PSV1**

Les modules SCA associés à cet instantané d'application de processus incluent le contexte de version, comme indiqué dans le tableau suivant :

Tableau 15. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-T1-PSV1-M1	PA1-T1-PSV1-M1.ear
M2	PA1-T1-PSV1-M2	PA1-T1-PSV1-M2.ear

Conventions de dénomination pour les déploiements de Process Server :

Sur le Process Server, vous pouvez déployer un instantané d'une application de processus. Le sigle de l'instantané d'application de processus est utilisé pour identifier de manière unique la version.

Pour les déploiements d'instantané d'application de processus, le contexte de la version combine les éléments suivants :

- Sigle du nom de l'application de processus
- Sigle de l'instantané d'application de processus

Les ressources doivent être nommées de manière unique et identifiées de façon externe avec le contexte de la version. Le tableau suivant contient des noms identifiés de manière unique :

Tableau 16. Exemples de nom et de sigle

Type de nom	Exemple
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Instantané d'application de processus	**1.0.0**
Sigle de l'instantané d'application de processus	**1.0.0**

Une ressource, par exemple un module ou une bibliothèque, contient le contexte de version dans son identifiant.

Le tableau suivant contient deux modules et illustre comment les fichiers EAR associés incluent le contexte de version.

Tableau 17. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

Le tableau suivant contient deux bibliothèques ayant pour portée l'application de processus. Il montre comment les fichiers JAR associés incluent le contexte de version.

Tableau 18. Bibliothèques ayant pour portée l'application de processus et fichiers JAR avec version

Nom de nom de bibliothèque ayant pour portée une application de processus SCA	Nom avec version	Nom de fichier JAR avec version
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Liaisons avec versions

Les applications de processus peuvent contenir des modules SCA contenant des liaisons d'importation et d'exportation. Lorsque vous codéployez des applications, la liaison de chaque version d'une application doit être unique. Certaines liaisons sont automatiquement mises à jour au cours du déploiement pour s'assurer que chacune des versions dispose d'une liaison unique. Dans d'autres cas, vous devez mettre à jour la liaison après le déploiement pour vous en assurer.

Une liaison *spécifique à une version* recouvre une version particulière d'une application de processus, ce qui garantit que chaque liaison est unique pour chacune des applications de processus. Les sections suivantes décrivent les liaisons qui sont automatiquement mises à jour avec une version spécifique ainsi que les actions que vous devez exécuter pendant la phase d'exécution dans le cas d'une liaison sans version spécifique. Pour plus d'informations sur les éléments à prendre en compte lorsque vous créez des modules, voir «Eléments à prendre en compte lors de l'utilisation de liaisons».

SCA

La cible d'une liaison SCA est automatiquement renommée si les liaisons d'importation et d'exportation du module sont définies dans la même portée d'application de processus.

Dans le cas contraire, un message d'information est consigné dans le fichier journal. Vous devez modifier la liaison d'importation après le déploiement pour changer l'adresse cible du noeud final. Vous pouvez utiliser la console d'administration pour ce faire.

Service Web (JAX-WS or JAX-RPC)

L'adresse cible du noeud final d'une liaison de service Web est automatiquement renommée avec une version spécifique au cours du déploiement quand toutes les conditions suivantes sont réunies :

- Vous avez suivi la convention de nom par défaut pour l'adresse :
  http://*ip*:*port*/*nom_module*Web/sca/*nom_exportation*
- L'adresse de noeud final est une adresse SOAP/HTTP.
- Les liaisons d'importation et d'exportation du module sont définies dans la même portée d'application de processus.

Si ces conditions ne sont pas satisfaites, un message d'information est consigné dans le fichier journal. L'action à exécuter dépend de la manière dont vous déployez l'application de processus.

- Si vous déployez conjointement votre application de processus, vous devez renommer manuellement l'adresse URL du noeud final SOAP/HTTP ou la file d'attente de destination SOAP/JMS de manière

qu'elle soit unique pour chacune des versions de l'application de processus. Vous pouvez utiliser la console d'administration après le déploiement pour modifier l'adresse cible du noeud final.
- Si vous déployez une unique version de l'application de processus, vous pouvez ignorer ce message

Pour codéployer un instantané de liaison de service Web JMS/SOAP, l'action à exécuter dépend de la manière dont vous déployez votre application de processus :
- Si l'exportation cible et l'importation sont dans la même application de processus, exécutez les opérations suivantes avant de publier l'application de processus sur le Process Center puis créez l'instantané :
  1. Changez l'adresse URL de noeud final de l'exportation. Vérifiez que la destination et la fabrique de connexions sont différentes.
  2. Changez l'adresse URL de noeud final de l'importation de manière qu'elle soit identique à celle que vous avez spécifiée pour l'exportation à l'étape précédente.
- Si l'exportation cible et l'importation sont dans des applications de processus différents, exécutez les opérations suivantes :
  1. Changez l'adresse URL de noeud final de l'exportation. Vérifiez que la destination et la fabrique de connexions sont différentes.
  2. Publiez l'application de processus sur le Process Center.
  3. Créez l'instantané.
  4. Déployez l'application de processus sur le Process Server.
  5. Utilisez la console d'administration WebSphere pour modifier l'adresse URL de noeud final de l'importation correspondante de manière qu'elle soit identique à celle que vous avez spécifiée pour l'exportation.

HTTP

L'adresse URL de noeud final d'une liaison HTTP est automatiquement renommée avec une version spécifique au cours du déploiement quand toutes les conditions suivantes sont réunies :
- Vous avez suivi la convention de nom par défaut pour l'adresse :

  http(s)://*ip*:*port*/*nom_module*Web/*chemin_contexte_export*
- Les liaisons d'importation et d'exportation du module sont définies dans la même portée d'application de processus.

Si ces conditions ne sont pas satisfaites, un message d'information est consigné dans le fichier journal. L'action à exécuter dépend de la manière dont vous déployez l'application de processus.
- Si vous déployez conjointement votre application de processus, vous devez renommer manuellement l'adresse URL de noeud final de manière qu'elle soit unique pour chacune des versions de l'application de processus. Vous pouvez utiliser la console d'administration après le déploiement pour modifier l'adresse cible du noeud final.
- Si vous déployez une unique version de l'application de processus, vous pouvez ignorer ce message

JMS et JMS générique

Les liaisons JMS génériques et les liaisons JMS système prennent automatiquement en compte la version.

Remarque : Pour les liaisons JMS génériques et les liaisons JMS définies par l'utilisateur, il n'y a pas de renommage automatique des liaisons au cours du déploiement pour que les liaisons soient spécifiques à une version. Dans le cas d'une liaison définie par l'utilisateur, pour différencier les versions de l'application de processus, vous devez renommer les attributs suivants :
- Configuration de noeud final
- File d'attente de destination de réception

- Nom du port d'écoute (si défini)

Définissez la destination d'envoi associée si vous modifiez le noeud final du module cible.

MQ/JMS et MQ

Il n'y a pas de renommage automatique des liaisons MQ/JMS et MQ au cours du déploiement pour qu'elles soient spécifiques à une version.

Pour différencier les versions de l'application de processus, vous devez renommer les attributs suivants :
- Configuration de noeud final
- File d'attente de destination de réception

Définissez la destination d'envoi associée si vous modifiez le noeud final du module cible.

EJB

Il n'y a pas de renommage automatique des liaisons EJB au cours du déploiement pour qu'elles soient spécifiques à une version.

Pour différencier les versions de l'application de processus, vous devez renommer les attributs des noms JNDI.

Notez que vous devez aussi mettre à jour les applications client de manière qu'elles utilisent les nouveaux noms JNDI.

EIS

Un adaptateur de ressources est automatiquement renommé pour référencer la version au cours du déploiement si le nom de ressource par défaut (***ModuleName*App:Adapter Description**) n'a pas été modifié.

Si le nom de ressource par défaut a été modifié, les noms des adaptateurs de ressources doivent être différents pour chaque version de l'application de processus.

Si les noms des adaptateurs de ressources ne sont pas différents, un message d'information est journalisé pendant le déploiement pour vous en informer. Vous pouvez renommer manuellement les adaptateurs de ressources après le déploiement à l'aide de la console d'administration.

Appel dynamique avec version

Vous pouvez configurer des composants de flux de médiation pour acheminer des messages vers des noeuds finaux qui sont déterminés dynamiquement lors de l'exécution. Quand vous créez le module de médiation, vous configurez la recherche de noeud final de façon à utiliser le routage avec version.

Si vous utilisez le style IBM_VRM (*<version>.<édition>.<modification*) pour l'instantané, vous pouvez exporter le fichier EAR de l'application de processus dans WSRR (WebSphere Service Registry and Repository). Quand vous créez le module de médiation, vous configurez ensuite la recherche de noeud final de façon à utiliser le routage avec version. Par exemple, vous sélectionnez **Renvoyer le noeud final correspondant à la dernière version compatible des services utilisant le module SCA** dans la zone **Règle de correspondance** puis vous sélectionnez **SCA** pour **Type de liaison**.

Les versions suivantes de l'application de processus sont déployées sur le serveur et sont publiées dans WSRR. La recherche de noeud final de l'application de processus appelle dynamiquement la dernière version compatible du noeud final de service.

Notez que vous pouvez également définir la cible dans l'en-tête SMOHeader, et la valeur peut être alors transportée par le message de demande.

Déploiement d'applications de processus contenant des projets et des modules Java

Les applications de processus peuvent contenir des projets Java et des modules Java EE personnalisés. Lorsque vous codéployez des applications, le module Java EE personnalisé de chaque version de l'application doit être unique.

Notez que les modules Java EE personnalisés et les projet Java sont déployés sur un serveur s'ils sont déployés avec un module SCA contenant une dépendance déclarée. Si vous ne sélectionnez pas **Déployer avec le module** (valeur par défaut) quand vous déclarez la dépendance, vous devez déployer le module ou le projet manuellement.

Déploiement d'applications de processus contenant des règles métier et des sélecteurs

Si vous déployez plusieurs versions d'une application de processus qui inclut un composant de sélecteur ou une règle métier, faites attention à la manière dont les métadonnées associées sont utilisées par les versions.

Les métadonnées dynamiques d'une règle métier ou d'un composant de sélecteur sont définies pendant la phase d'exécution par le nom de composant, l'espace de nom cible du composant et le type de composant. Si vous déployez au moins deux versions d'une application de processus qui contient une règle métier ou un sélecteur dans le même environnement d'exécution, elles partageront les mêmes métadonnées de logique de règle (règle métier) ou de routage (sélecteur).

Pour que chaque version de la règle métier ou du composant de sélecteur de l'application de processus utilise ses propres métadonnées dynamiques (logique de règle ou routage), restructurez l'espace de nom cible pour qu'ils soit spécifique à chaque version de l'application de processus.

Objets de configuration

Vous pouvez utiliser les commandes AdminConfig de l'outil d'administration de ligne de commande WebSphere (wsadmin) pour afficher et modifier les propriétés de sécurité et de la base de données dans IBM Business Process Manager.

Le terme *objet de configuration* se réfère à un objet auquel on accède à l'aide des commandes AdminConfig de wsadmin. Pour de plus amples informations, voir Propriétés de configuration de la sécurité.

Architecture de déploiement

L'architecture de déploiement de IBM Business Process Manager est constituée de processus logiciels appelés serveurs, d'unités topologiques référencées comme noeuds et cellules et du référentiel de configuration utilisé pour stocker les informations de configuration.

Cellules

Dans IBM Business Process Manager, les *cellules* sont des regroupements logiques d'un ou plusieurs noeuds dans un réseau réparti.

Une cellule est un concept de configuration, un moyen pour les administrateurs d'associer logiquement des noeuds entre eux. Les administrateurs définissent les noeuds qui constituent une cellule, en fonction de critères spécifiques qui ont une signification dans leur environnement organisationnel.

Les données de configuration d'administration sont stockées dans des fichiers XML. Une cellule conserve des fichiers de configuration maîtres pour chaque serveur de chaque noeud de la cellule. Chaque noeud et chaque serveur possèdent également leurs propres fichiers de configuration locaux. Les modifications apportées à un noeud local ou à un fichier de configuration de serveur sont temporaires, si le serveur

appartient à la cellule. Tant qu'elles sont en vigueur, les modifications locales remplacent les configurations de cellule. Les modifications apportées aux fichiers de configuration du serveur maître et du noeud maître au niveau de la cellule remplacent les modifications temporaires effectuées sur le noeud lorsque les documents de configuration de la cellule sont synchronisés avec les noeuds. La synchronisation a lieu lors d'événements désignés (par exemple, au démarrage d'un serveur).

Serveurs

Les serveurs assurent les fonctions centrales de IBM Business Process Manager. Les serveurs de processus permettent d'étendre, ou d'accroître, les capacités d'un serveur d'applications à gérer les modules SCA (Service Component Architecture). D'autres serveurs (gestionnaires de déploiement et agents de noeud) permettent de gérer les serveurs de processus.

Un serveur de processus peut être un *serveur autonome* ou un *serveur géré*. Un serveur géré peut être membre d'un *cluster*. L'ensemble des serveurs gérés, des clusters de serveurs et des autres logiciels intermédiaires constitue un *environnement de déploiement*. Dans un environnement de déploiement, chaque serveur géré ou cluster est configuré pour assurer une fonction particulière dans l'environnement de déploiement (par exemple, hôte de destination, hôte de module d'application ou serveur CEI (Common Event Infrastructure). Un serveur autonome est configuré de façon à assurer toutes les fonctions nécessaires.

Les serveurs assurent le fonctionnement de l'environnement d'exécution des modules SCA des ressources utilisées par ces modules (sources de données, spécifications d'activation et destinations JMS) et des ressources fournies par IBM (destinations de messages, conteneurs Business Process Choreographer et serveurs CEI (Common Event Infrastructure).

Un *agent de noeud* est un agent d'administration qui représente un noeud sur votre système et gère les serveurs de ce noeud. Les agents de noeud surveillent les serveurs d'un système hôte et acheminent les demandes d'administration vers les serveurs. Un agent de noeud est créé lorsqu'un noeud est fédéré sur un gestionnaire de déploiement.

Un *gestionnaire de déploiement* est un agent d'administration qui fournit une vue centralisée de la gestion aux différents serveurs et clusters.

Un serveur autonome est défini par un profil autonome, un gestionnaire de déploiement est défini par un profil de gestionnaire de déploiement et les serveurs gérés sont créés au niveau d'un *noeud géré*, qui est défini par un profil de noeud géré.

Serveurs autonomes :

Un serveur autonome fournit un environnement pour le déploiement de modules SCA dans un processus serveur unique. Ce processus serveur inclut, entre autres, une console d'administration, une cible de déploiement, le support de messagerie, le gestionnaire de règles métier et un serveur CEI (Common Event Infrastructure).

Un serveur autonome est simple à configurer. Il est équipé d'une console de démarrage rapide permettant de démarrer et d'arrêter le serveur, ou encore d'ouvrir la galerie d'exemples et la console d'administration. Si vous installez les exemples d'IBM Business Process Manager, puis ouvrez la galerie d'exemples, un exemple de solution est déployé sur le serveur autonome. Vous pouvez explorer les ressources utilisées pour cet exemple dans la console d'administration.

Vous pouvez déployer vos propres solutions sur un serveur autonome, mais celui-ci ne dispose pas de la capacité, de l'évolutivité ni de la robustesse nécessaires dans un environnement de production. L'utilisation d'un environnement de déploiement réseau est préférable en environnement de production.

Il est possible de commencer par utiliser un serveur autonome, puis d'inclure celui-ci dans un environnement de déploiement réseau en le fédérant à une cellule de gestionnaire de déploiement, *sous*

réserve qu'aucun autre noeud n'ait été fédéré avec cette cellule. Il n'est pas possible de fédérer plusieurs serveurs autonomes dans une seule cellule. Pour fédérer le serveur autonome, utilisez soit la console d'administration du gestionnaire de déploiement, soit la commande **addNode**. Le serveur autonome ne doit pas être en cours d'exécution lorsque vous le fédérez au moyen de la commande **addNode**.

Un serveur autonome est défini par un profil de serveur autonome.

Clusters :

Les clusters sont des groupes de serveurs qui sont gérés ensemble et participent à la gestion de la charge de travail.

Un cluster peut contenir des noeuds ou des serveurs d'applications individuels. Un noeud correspond généralement à un système informatique physique possédant une adresse hôte IP distincte, qui exécute un ou plusieurs serveurs d'applications. Les clusters peuvent être regroupés sous la configuration d'une cellule, qui associe logiquement entre eux de nombreux serveurs et clusters possédant des configurations et des applications différentes en fonction de critères choisis par l'administrateur en adéquation avec ses environnements organisationnels.

Les clusters sont chargés d'équilibrer la charge de travail entre les serveurs. Les serveurs qui appartiennent à un cluster sont appelés membres du cluster. Lorsque vous installez une application sur un cluster, cette application est automatiquement installée sur chaque membre du cluster.

Chaque membre du cluster contenant les mêmes applications, vous pouvez distribuer les tâches client en fonction des capacités des différentes machines, en affectant des pondérations à chaque serveur.

L'affectation de pondérations aux serveurs d'un cluster améliore les performances et la reprise sur incident. Les tâches sont affectées au serveurs qui disposent de la capacité requise pour effectuer les opérations associées. Si un serveur n'est pas disponible pour effectuer la tâche, cette dernière est affectée à un autre membre du cluster. La possibilité de réaffecter les tâches offre des avantages évidents par rapport à l'exécution d'un seul serveur d'applications, qui peut vite être surchargé lorsque le nombre de demandes est trop important.

Profils

Un profil définit un environnement d'exécution unique, associé à des fichiers spécifiques (commandes, configuration et journaux). Les profils définissent trois types d'environnement différents sur les systèmes IBM Business Process Manager : serveur autonome, gestionnaire de déploiement et noeud géré.

Les profils permettent de définir plusieurs environnements d'exécution sur un système sans installer plusieurs copies des fichiers binaires d'IBM Business Process Manager.

Utilisez l'Utilitaire de ligne de commande BPMConfig pour créer des profils IBM BPM. L'utilitaire de ligne de commande **manageprofiles** ou son interface utilisateur graphique, Profile Management Tool (PMT), peuvent être utilisés comme alternative pour créer le profil de gestionnaire de déploiement ou le profil de noeud géré. PMT n'est plus pris en charge pour la création de profils autonomes.

Remarque : Sur les plateformes réparties, chaque profil possède un nom unique. Sous z/OS, tous les profils sont nommés «default». Vous ne pouvez pas renommer, modifier, copier ou supprimer des profils sous z/OS.

Types de profils

Les types de profils IBM BPM suivants sont disponibles avec IBM Business Process Manager version 8.5 :

Profil autonome IBM BPM
Le profil autonome définit des serveurs autonomes avec des capacités et fonctionnalités

spécifiques aux configurations d'IBM BPM Express. Vous pouvez créer le profil autonome à l'aide du modèle de profil fourni sous `BPM/BpmServer`, qui est uniquement pris en charge dans IBM BPM Express.

Profil de gestionnaire de déploiement IBM BPM
Le profil de gestionnaire de déploiement définit un gestionnaire de déploiement fournissant une interface d'administration unique à un groupe logique de serveurs sur un ou plusieurs postes de travail. Vous pouvez créer le profil de gestionnaire de déploiement à l'aide du modèle de profil fourni sous `BPM/BpmDmgr`, qui est uniquement pris en charge dans IBM BPM Standard et IBM BPM Advanced.

Profil de noeud géré IBM BPM
Le profil de noeud géré définit un noeud géré lorsque le noeud est fédéré à un gestionnaire de déploiement. Vous pouvez créer le profil de noeud géré à l'aide du modèle de profil fourni sous `BPM/BpmNode`, qui est uniquement pris en charge dans IBM BPM Standard et IBM BPM Advanced.

Les types de profils spécifiques sont disponibles pour chaque configuration IBM BPM.

Tableau 19. Types de profils IBM BPM disponibles

	Types de profils		
Configuration IBM BPM	**Autonome**	**Gestionnaire de déploiement**	**Noeud géré**
IBM BPM Express	Oui	Non	Non
IBM BPM Standard	Non	Oui	Oui
IBM BPM Advanced	Non	Oui	Oui
Environnement UTE (Unit Test Environment) pour IBM Integration Designer	Oui	Facultatif	Facultatif

Répertoire de profil

Chaque profil du système possède son propre répertoire contenant tous ses fichiers. Vous indiquez l'emplacement de ce répertoire lors de la création du profil. Par défaut, il s'agit du répertoire `profiles` dans le répertoire où IBM Business Process Manager est installé. Par exemple, le profil Dmgr se trouve sous `C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr`.

Profil par défaut

Le premier profil créé sur une installation d'IBM Business Process Manager représente le *profil par défaut*. Ce profil est la cible par défaut des commandes émises à partir du répertoire `bin` situé dans le répertoire racine d'installation d'IBM Business Process Manager. Lorsqu'il n'existe qu'un seul profil sur un système, chaque commande fonctionne sur ce profil. Si vous créez un autre profil, vous pouvez faire de ce profil le profil par défaut.

Remarque : Le profil par défaut n'est pas nécessairement un profil portant le nom «par défaut».

Extension de profils

Si vous avez déjà un profil de gestionnaire de déploiement, un profil de noeud géré ou un profil serveur autonome créé pour WebSphere Application Server Network Deployment, vous pouvez l'*étendre* pour prendre en charge IBM Business Process Manager en plus de la fonction existante. Pour étendre un profil, vous devez tout d'abord installer IBM Business Process Manager. Utilisez ensuite l'utilitaire de ligne de commande **manageprofiles** pour étendre un profil autonome ou utilisez le Profile Management Tool ou l'utilitaire de ligne de commande **manageprofiles** pour étendre un profil de gestionnaire de déploiement ou un profil de noeud géré.

Important : Dans un environnement de déploiement réseau, vous devez d'abord étendre le profil de gestionnaire de déploiement, puis les profils de noeud géré.

Restriction : Vous ne pouvez pas étendre un profil autonome ou un profil de gestionnaire de déploiement si le registre d'utilisateur par défaut de WebSphere VMM a été modifié, par exemple, pour utiliser LDAP.

Gestionnaires de déploiement

Un gestionnaire de déploiement est un serveur permettant de gérer les opérations liées à un groupe logique ou à une cellule comprenant d'autres serveurs. Le gestionnaire de déploiement est l'emplacement central permettant d'administrer les serveurs et clusters.

Lors de la création d'un environnement de déploiement, le profil du gestionnaire de déploiement est le premier profil créé. Chaque environnement de déploiement créé possède une console de démarrage rapide permettant de démarrer et arrêter le gestionnaire de déploiement et de démarrer la console d'administration de celui-ci. La console d'administration du gestionnaire de déploiement permet de gérer les serveurs et clusters contenus dans la cellule. Ces opérations incluent la configuration des serveurs et des clusters, l'ajout de serveurs à des clusters et le déploiement de modules SCA.

Le gestionnaire de déploiement est lui-même un serveur, mais vous ne pouvez pas y déployer de modules.

Noeuds

Un *noeud* est un regroupement logique de serveurs gérés.

Un noeud correspond généralement à un système informatique logique ou physique doté d'une adresse hôte IP distincte. Les noeuds ne peuvent pas couvrir plusieurs ordinateurs. Les noms de noeud sont généralement identiques au nom d'hôte de l'ordinateur.

Les noeuds d'une topologie de déploiement réseau peuvent être gérés ou non gérés. Un noeud géré possède un processus d'agent de noeud qui gère sa configuration et ses serveurs. Les noeuds non gérés ne possèdent pas d'agent de noeud.

Noeuds gérés :

Un *noeud géré* est un noeud qui est fédéré dans un gestionnaire de déploiement et qui contient un agent de noeud et peut contenir des serveurs gérés. Sur un noeud géré, vous pouvez configurer et exécuter des serveurs gérés.

Les serveurs configurés sur un noeud géré constituent les ressources de votre environnement de déploiement. Les opérations de création, configuration, démarrage, arrêt, gestion et suppression de ces serveurs s'effectuent via la console d'administration du gestionnaire de déploiement.

Une noeud géré est un agent de noeud qui gère tous les serveurs sur un noeud.

Lorsqu'un noeud est fédéré, un processus d'agent de noeud est créé automatiquement. Cet agent de noeud doit être en cours d'exécution pour permettre de gérer la configuration du profil. Par exemple, lorsque vous effectuez les tâches suivantes :

- Démarrer et arrêter les processus serveur.
- Synchroniser les données de configuration sur le gestionnaire de déploiement avec la copie située sur le noeud.

Toutefois, l'agent de noeud ne doit pas nécessairement être en cours d'exécution pour que les applications soient exécutées, ni pour que les ressources soient configurées sur le noeud.

Un noeud géré peut contenir un ou plusieurs serveurs, qui sont administrés par un gestionnaire de déploiement. Vous pouvez déployer des solutions sur les serveurs en mode géré, mais le noeud géré ne contient pas de galerie d'exemples d'applications. Le noeud géré est défini par un profil géré par noeud et possède une console de démarrage rapide.

Noeuds non gérés :

Un noeud non géré ne possède pas d'agent de noeud pour gérer ses serveurs.

Des noeuds non gérés dans la topologie de déploiement réseau peuvent avoir des définitions de serveurs, comme des serveurs Web, mais pas de définitions de serveur d'applications. Les noeuds non gérés ne peuvent jamais être fédérés. Cela signifie qu'un agent de noeud ne peut jamais être ajouté à un noeud non géré. Un serveur autonome représente un autre type de noeud non géré. Le gestionnaire de déploiement ne peut pas gérer ce serveur autonome car il n'est pas connu de la cellule. Un serveur autonome peut être fédéré. Lorsqu'il est fédéré, un agent de noeud est automatiquement créé. Le noeud devient un noeud géré dans la cellule.

Agents de noeud

Les agents de noeud sont des agents d'administration qui acheminent les demandes administratives destinées aux serveurs.

Un agent de noeud est un serveur exécuté sur chaque système informatique hôte qui participe à la configuration de Network Deployment. Il s'agit purement d'un agent d'administration, qui n'est pas impliqué dans les fonctions de traitement des applications. Un agent de noeud héberge également d'autres fonctions d'administration importantes, telles que les services de transfert de fichiers, la synchronisation de la configuration et le contrôle des performances.

Remarques relatives aux noms de profils, de noeuds, de serveurs, d'hôtes et de cellules

Cette rubrique indique les termes réservés et les conditions à respecter pour nommer un profil, un noeud, un serveur, un hôte et une cellule (le cas échéant). Cette rubrique s'applique aux plateformes réparties.

Remarques relatives aux noms de profils

Le nom de profil peut être tout nom unique, avec les restrictions suivantes. N'utilisez aucun des caractères suivants :

- Espaces
- Caractères spéciaux non autorisés dans un nom de répertoire sur le système d'exploitation, par exemple *, & ou ?.
- Barres obliques (/) ou barres obliques inversées (\)

Les caractères codés sur deux octets sont autorisés.

Windows **Remarques liées au chemin de répertoire :** Le chemin du répertoire d'installation ne doit pas comporter plus de 60 caractères. Le nombre de caractères du répertoire *chemin_répertoire_profils\nom_profil* doit être inférieur ou égal à 80 caractères.

Remarque : Utilisez une convention de dénomination de chemin court lorsque vous créez un profil dans l'environnement Windows pour éviter la limitation à 255 caractères des chemins Windows.

Remarques relatives aux noms de noeuds, de serveurs, d'hôtes et de cellules

Noms réservés : Evitez d'utiliser des noms réservés comme valeurs de zones. En effet, l'utilisation de noms réservés peut entraîner des résultats imprévisibles. Les mots suivants sont réservés :

- cellules

- noeuds
- serveurs
- clusters
- applications
- déploiements

Descriptions des zones figurant dans les pages Noms de noeud et d'hôte et Noms de noeud, d'hôte et de cellule : suivez les instructions d'attribution de noeud appropriées lorsque vous créez les profils.

- Profils de serveur autonomes
- Profils du gestionnaire de déploiement
- Profils de noeud géré

Tableau 20. Instructions d'attribution de nom pour les profils de serveur autonomes

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom du noeud	Linux UNIX Windows *shortHostName* Node *NodeNumber* où : • *shortHostName* représente le nom d'hôte abrégé. • *NodeNumber* est un nombre séquentiel commençant, séquence commençant à partir de 01.	N'utilisez pas de nom réservé.	Sélectionnez le nom de votre choix. Si vous envisagez de créer plusieurs serveurs sur le même système, choisissez un nom unique afin de simplifier l'installation.
Nom du serveur	Linux UNIX Windows server1	Utilisez un nom unique pour le serveur.	Nom logique du serveur.
Nom d'hôte	Linux UNIX Windows Nom long du serveur DNS (Domain Name Server).	Utilisez un nom de système hôte qualifié complet dont l'adresse peut être résolue sur le réseau.	Utilisez le nom DNS ou l'adresse IP du poste de travail pour permettre la communication avec ce dernier. Consultez les informations supplémentaires sur le nom d'hôte, à la suite de ce tableau.

Tableau 21. Instructions d'attribution de nom pour les profils de gestionnaire de déploiement

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom du noeud	Linux UNIX Windows *shortHostName* Cell Manager*Node Number* où : • *shortHostName* représente le nom d'hôte abrégé. • *NodeNumber* est un nombre séquentiel commençant, séquence commençant à partir de 01.	Utilisez un nom unique pour le gestionnaire de déploiement. N'utilisez pas de nom réservé.	Le nom est utilisé à des fins d'administration dans la cellule de gestionnaire de déploiement.
Nom d'hôte	Linux UNIX Windows Nom long du serveur DNS (Domain Name Server).	Utilisez un nom de système hôte qualifié complet dont l'adresse peut être résolue sur le réseau. N'utilisez pas de nom réservé.	Utilisez le nom DNS ou l'adresse IP du poste de travail pour permettre la communication avec ce dernier. Consultez les informations supplémentaires sur le nom d'hôte, à la suite de ce tableau.

Tableau 21. Instructions d'attribution de nom pour les profils de gestionnaire de déploiement (suite)

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom de la cellule	Linux UNIX Windows *shortHostName* Cell *CellNumber* où : • *shortHostName* représente le nom d'hôte abrégé. • *CellNumber* représente un chiffre séquentiel commençant par 01.	Utilisez un nom unique pour la cellule du gestionnaire de déploiement. Un nom de cellule doit être unique dans tous les cas où le produit s'exécute sur le même poste de travail physique ou le même cluster de postes de travail, par exemple un Sysplex. En outre, il doit être unique dès lors que la connectivité réseau entre entités est requise entre les cellules ou à partir d'un client devant communiquer avec chacune des cellules. Les noms de cellule doivent également être uniques si les espaces noms associés sont sur le point d'être fédérés. Si cette condition n'est pas respectée, des erreurs de type javax.naming.Name NotFoundException peuvent survenir, au quel cas vous devez créer des cellules avec des noms uniques.	Tous les noeuds fédérés deviennent membres de la cellule du gestionnaire de déploiement définie dans la page des noms de noeud, d'hôte et de cellule de l'outil de gestion des profils.

Tableau 22. Instructions d'attribution de nom pour les profils de noeud géré

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom du noeud	Linux UNIX Windows *shortHostName* Node *NodeNumber* où : • *shortHostName* représente le nom d'hôte abrégé. • *NodeNumber* est un nombre séquentiel commençant, séquence commençant à partir de 01.	N'utilisez pas de nom réservé. Utilisez un nom unique dans la cellule du gestionnaire de déploiement.	Ce nom est utilisé à des fins d'administration dans la cellule de gestionnaire à laquelle le profil de noeud géré est ajouté. Utilisez un nom unique dans la cellule du gestionnaire de déploiement.
Nom d'hôte	Linux UNIX Windows Nom long du serveur DNS (Domain Name Server).	Utilisez un nom de système hôte qualifié complet dont l'adresse peut être résolue sur le réseau.	Utilisez le nom DNS ou l'adresse IP du poste de travail pour permettre la communication avec ce dernier. Consultez les informations supplémentaires sur le nom d'hôte, à la suite de ce tableau.

Remarques concernant le nom d'hôte :

Le nom d'hôte correspond au nom réseau du poste de travail physique sur lequel le noeud est installé. Il doit être résolu en noeud réseau physique sur le serveur. Si le serveur contient plusieurs cartes réseau, le nom d'hôte ou l'adresse IP doit être résolu sur l'une d'elles. Les noeuds distants utilisent le nom d'hôte pour se connecter à ce noeud et communiquer avec lui.

IBM Business Process Manager est compatible avec le protocole IP version 4 (IPv4) et version 6 (IPv6). Chaque fois que des adresses IP peuvent être indiquées dans la console d'administration ou via un autre point d'accès, vous pouvez spécifier l'un ou l'autre format. Il est à noter que si le protocole IPv6 est mis en oeuvre sur votre système, vous devez spécifier l'adresse IP suivant ce format. Inversement, si ce protocole n'est pas disponible, entrez les adresses IP au format IPv4. Pour plus d'informations sur IPv6, reportez-vous à la description suivante : IPv6.

Les instructions suivantes peuvent aider à déterminer le nom d'hôte approprié à appliquer à votre poste de travail :

- Sélectionnez un nom d'hôte accessible via les autres postes de travail du réseau.
- N'utilisez pas l'identificateur générique 'localhost' pour cette valeur.
- Ne tentez pas d'installer les produits IBM Business Process Manager sur un serveur portant un nom d'hôte qui utilise des caractères DBCS (Double-Byte Character Set). En effet, les caractères DBCS ne sont pas pris en charge lorsqu'ils sont utilisés dans le nom d'hôte.
- Evitez d'utiliser le trait de soulignement (_) dans les noms de serveurs. Les normes Internet exigent que les noms de serveurs soient conformes aux normes décrites dans les documents Internet Official Protocol Standards RFC 952 et RFC 1123. Les noms de domaines ne doivent contenir que des lettres (en majuscules et en minuscules) et des chiffres. Les noms de domaines peuvent également contenir des tirets (-), sous réserve de ne pas se trouver en fin de nom. Les traits de soulignement (_) ne sont pas acceptés dans le nom d'hôte. Si vous avez installé IBM Business Process Manager sur un serveur dont le nom comporte un trait de soulignement, vous devez accéder à ce serveur au moyen de son adresse IP jusqu'à ce que vous l'ayez renommé.

Si vous définissez des noeuds coexistant sur le même système avec des adresses IP uniques, définissez chaque adresse IP dans une table de recherche DNS (Domain Name Server). Les fichiers de configuration des serveurs ne fournissent pas de fonction de résolution du nom de domaine pour les adresses IP définies sur un poste de travail doté d'une adresse réseau unique.

La valeur indiquée pour le nom d'hôte est utilisée pour la propriété hostName dans les documents de configuration. Indiquez la valeur du nom d'hôte dans l'un des formats suivants :

- Chaîne représentant le nom d'hôte DNS (Domain Name Server) complet, tel que xmachine.manhattan.ibm.com
- Nom d'hôte DNS abrégé par défaut, tel que xmachine
- Adresse IP numérique, telle que 127.1.255.3

Le nom d'hôte DNS complet permet d'éviter toute ambiguïté et est extrêmement souple. Vous avez la possibilité de modifier l'adresse IP réelle du système hôte sans modifier la configuration du serveur. La valeur définie pour le nom d'hôte est particulièrement utile si vous avez l'intention de modifier fréquemment l'adresse IP lorsque vous utilisez DHCP (Dynamic Host Configuration Protocol) pour affecter des adresses IP. L'inconvénient de ce format est qu'il dépend d'un serveur DNS. Si le serveur DNS n'est pas disponible, la connectivité est compromise.

Le nom d'hôte peut également être résolu de manière dynamique. En outre, le format de nom court étant redéfini dans le fichier hosts local, le système peut exécuter le serveur, même si ce dernier est déconnecté du réseau. Associez le nom abrégé à la valeur 127.0.0.1 (boucle locale) dans le fichier hosts pour lancer l'exécution en étant déconnecté. L'inconvénient du format de nom abrégé est qu'il dépend d'un serveur DNS pour l'accès distant. Si le serveur DNS n'est pas disponible, la connectivité est compromise.

Dans ce dernier cas, la résolution du nom via DNS n'est pas nécessaire. Un noeud distant peut se connecter à l'hôte désigné par une adresse IP sans avoir recours au serveur DNS. L'inconvénient de ce format est que l'adresse IP numérique est fixe. Vous devez modifier la propriété hostName dans les fichiers de configuration lorsque vous modifiez l'adresse IP du poste de travail. Par conséquent, n'utilisez pas d'adresse IP si vous utilisez le protocole DHCP (Dynamic Host Configuration Protocol) ou si vous changez souvent d'adresse IP. En outre, vous ne pouvez pas utiliser le noeud si l'adresse IP de l'hôte est déconnectée du réseau.

BPMN 2.0

IBM Business Process Manager prennent en charge la sous-classe Common Executable de la classe de conformité de BPMN 2.0 Process Modeling, qui traite des modèles exécutables.

BPMN (Business Process Model and Notation) est la norme de base des processus dans IBM Process Designer et IBM Process Center. Les diagrammes de définition de processus métier reposent sur la spécification BPMN. Cette rubrique présente certaines des méthodes par lesquelles BPMN 2.0 est appliqué dans IBM Business Process Manager. Pour obtenir des informations sur BPMN, reportez-vous à la page des spécifications BPMN à l'adresse suivante http://www.bpmn.org/.

IBM Business Process Manager prend en charge les types de tâche BPMN 2.0 suivants :

- Aucun (tâche abstraite dans la spécification BPMN 2.0)
- Tâche système (tâche de maintenance dans la spécification BPMN 2.0)
- Tâche utilisateur
- Script
- Tâche de décision (tâche de règle métier dans la spécification BPMN 2.0)

Les événements de message intermédiaires IBM BPM fournissent des fonctionnalités similaires à la tâche d'envoi et de réception BPMN.

Notation BPMN 2.0

A partir de la V7.5.1, les nouvelles icônes Process Designer de BPMN 2.0 des diagrammes de BPD sont rassemblées dans une palette simplifiée et affichées dans les diagrammes de processus. Ces icônes indiquent si votre activité est une tâche système, une tâche utilisateur, une tâche de décision, un script ou un processus lié. Les activités des modèles créés dans des versions antérieures affichent également les types de tâches et les icônes de tâches BPMN 2.0 appropriées lorsque vous les affichez dans la version 7.5.1 ou ultérieure.

Activités et tâches

Certaines modifications de terminologie ont été effectuées par rapport aux versions précédentes de Process Designer. Un certain nombre d'entre elles concernent des types d'activités qui ont été renommés.

- Les activités de maintenance (automatisées) sont à présent des tâches système.
- Les activités de maintenance (tâche) d'un couloir non système sont à présent des tâches utilisateur.
- Les activités de maintenance (tâche) d'un couloir système sont à présent des tâches de décision si elles font référence à un service de décision.
- Les activités de maintenance (tâche) d'un couloir système sont à présent des tâches système si elles font référence à un type de service autre qu'un service de décision.
- Les activités Javascript sont à présent des tâches de script.
- Les activités de processus imbriquées sont à présent des processus liés.
- Les activités externes provenant des versions précédentes de Process Designer sont disponibles en tant qu'implémentations externes pour les tâches utilisateur ou les tâches système.

Passerelles

Il n'existe pas de changements de notation concernant les passerelles des versions précédentes. Il existe toutefois trois changements de terminologie. La passerelle de décision est à présent la *passerelle exclusive*, la passerelle de jointure/de division simple est à présent la *passerelle parallèle* et la passerelle de jointure/de division conditionnelle est à présent la *passerelle inclusive*.

Il existe également un nouveau type de passerelle, appelé la *passerelle d'événement*. Une passerelle de branchement représente un point de branchement dans un processus dans lequel les chemins de substitution qui suivent la passerelle sont basés sur des événements qui se produisent, plutôt que sur l'évaluation des expressions utilisant des données de processus (comme avec une passerelle exclusive ou inclusive). Un événement spécifique, généralement le reçu d'un message, détermine le chemin qui sera pris.

Evénements sans interruption

BPMN 2.0 a ajouté une notation pour les événements sans interruption. Par défaut, un événement limite interrompt l'activité à laquelle il est attaché. Lorsque l'événement est déclenché, l'activité est arrêtée et le jeton poursuit son chemin dans le flux de séquence sortant de l'événement. Si l'événement est défini comme sans interruption, lorsqu'il est déclenché, l'activité attachée continue en parallèle et un nouveau jeton est généré puis transmis au flux de séquence sortant de l'événement. La limite de l'événement se transforme en ligne en pointillés pour les événements sans interruption.

Les événements intermédiaires attachés aux activités sont à présent des événements intermédiaires d'interruption s'ils ferment leurs activités attachées, ou des événements intermédiaires sans interruption dans le cas contraire.

Evénement de démarrage

La spécification BPMN permet aux modèles de processus d'omettre les symboles d'événement de démarrage et de fin. Process Designer nécessite que les modèles de processus utilisent des événements de démarrage et de fin.

Trois types d'événements de début sont disponibles dans Process Designer :

processus

- aucun
- message
- ad hoc

sous-processus

- aucun

sous-processus d'événement

- erreur
- message
- minuteur

Vous pouvez modifier le type d'un événement de démarrage en éditant les propriétés de l'événement. Un processus peut comporter plusieurs événements de démarrage de message mais vous ne pouvez utiliser qu'un seul événement de démarrage Aucun.

Les événements de fin

Il existe quatre types d'événement de fin disponibles : *message*, *terminer*, *erreur* et *aucun*. Vous pouvez changer le type d'un événement de fin.

Lorsqu'un processus parent appelle un processus enfant et que ce processus enfant exécute une action d'événement Terminer, le processus enfant s'arrête et le processus parent passe alors aux étapes suivantes.

Sous-processus

La spécification BPMN définit deux types de sous-processus, imbriqué et réutilisable. Vous pouvez créer les deux types dans Process Designer. Les sous-processus imbriqués sont appelés simplement *sous-processus* dans Process Designer et sont nouveaux dans la version 7.5.1. Le sous-processus réutilisable BPMN est appelé *processus lié* dans Process Designer.

Un sous-processus existe dans le processus contenant. Il permet de rassembler des étapes du processus dans le but de réduire la complexité et l'encombrement des diagrammes. Les sous-processus regroupent plusieurs étapes en une seule activité. Le sous-processus ne peut être vu que par le processus dans lequel il est défini. Un sous-processus existe dans l'étendue de son appelant et il a accès à toutes les variables contenues dans cet environnement. Il n'existe aucun paramètre transmis à partir de ou à destination du sous-processus intégré.

En dehors du sous-processus et du processus lié, Process Designer comporte un sous-processus d'événement, qui est un sous-processus spécialisé utilisé pour la gestion des événements. Il n'est pas relié à d'autres activités via le flux de séquences et est exécuté uniquement si son événement de démarrage est déclenché.

Processus liés

Un sous-processus réutilisable BPMN est appelé *processus lié* dans Process Designer. Il s'agit d'un processus créé en dehors du processus en cours pouvant être appelé par le processus actuel. IL est réutilisable car les autres définitions de processus peuvent également appeler ce processus. Le processus lié définit ses paramètres d'entrée et de sortie et il n'a pas accès à la portée ou à l'environnement de l'appelant. Le processus lié est similaire au processus imbriqué qui était disponible dans les anciennes versions ; aucun changement de comportement de l'activité n'est à noter. Les processus imbriqués existants sont migrés vers les processus liés. Le processus lié ressemble à un sous-processus avec une limite épaisse et il est mis en évidence dans la fenêtre Inspecteur.

Boucles

BPMN offre le concept d'une activité pouvant être répétée. L'activité peut être atomique, ce qui signifie qu'elle peut être répétée, ou elle peut être un sous-processus encapsulant une série d'étapes qui sont répétées. Si vous développez l'activité répétée, vous voyez les activités contenues devant être exécutées de façon répétitive. La condition est toujours évaluée au début de chaque itération de boucle. Il n'existe pas de possibilité d'évaluation à la fin de chaque itération de boucle.

IBM Business Process Manager possède une *boucle multi-instance*, qui est exécutée un nombre infini de fois avec les activités qu'elle contient de façon séquentielle ou en parallèle.

Importation de processus non-BPMN

Vous pouvez importer des modèles créés dans IBM WebSphere Business Modeler et les utiliser dans Process Designer. Pour plus d'informations sur l'importation BPMN 2.0, voir Mappage des éléments IBM WebSphere Business Modeler vers des constructions IBM Business Process Manager. Vous pouvez également importer des modèles BPMN 2.0 créés dans IBM WebSphere Business Compass, Rational Software Architect ou dans d'autres environnements de modélisation.

Définitions de processus métier (BPD)

Pour modéliser un processus dans IBM Process Designer, vous devez créer une définition de processus métier (BPD). La définition de processus métier peut se baser sur un modèle BPMN importé.

Une définition de processus métier est un modèle réutilisable d'un processus définissant ce qui est commun à toutes les instances d'exécution de ce modèle de processus. Une BPD doit contenir un événement de début, un événement de fin, au moins un couloir et une ou plusieurs activités. Pour plus d'informations sur les limitations de caractères qui s'appliquent au BPD, voir "Conventions de dénomination d'IBM Process Designer" dans les liens connexes.

Une définition de processus métier (BPD, Business Process Definition) doit inclure un couloir pour chaque système ou groupe d'utilisateurs qui participent à un processus. Un couloir peut être un couloir participant ou un couloir système. Toutefois, vous pouvez créer une BPD qui regroupe les activités d'un groupe et d'un système dans un même couloir si cela vous arrange. Pour plus d'informations sur la création d'une BPD, voir "Création d'une définition de processus métier (BPD)" dans les liens connexes.

Vous pouvez désigner n'importe quelle personne ou n'importe quel groupe comme responsable des activités d'un couloir participant. Chaque couloir que vous créez est affecté par défaut au groupe Tous les utilisateurs. Vous pouvez utiliser ce groupe par défaut pour exécuter et tester votre définition de processus métier dans Inspector. Le groupe Tous les utilisateurs inclut tous les utilisateurs membres du groupe de sécurité **tw_allusers**, lequel est un groupe de sécurité spécial incluant automatiquement tous les utilisateurs du système.

Un couloir système contient les activités gérées par un système IBM Process Center spécifique. Chaque activité nécessite une implémentation, qui définit l'activité et les propriétés de la tâche. Lors de l'implémentation, un développeur crée un service ou écrit le code JavaScript nécessaire à l'exécution des activités dans le couloir système. Pour plus d'informations sur les services, voir "Comprendre les types de services" dans les liens connexes.

Pour chaque définition de processus métier que vous créez, vous devez déclarer des variables afin de capturer les données métier qui sont transmises d'une activité à l'autre tout au long du processus. Pour plus d'informations sur l'implémentation des variables, voir "Gestion et mappage de variables" dans les liens connexes.

Vous pouvez également ajouter des événements à une BPD. Les événements présents dans IBM BPM peuvent être déclenchés par un dépassement de date d'exigibilité, par une exception ou par un message entrant. Le déclencheur souhaité détermine le type d'événement à implémenter. Pour plus d'informations sur les types d'événement disponibles et leurs déclencheurs, voir "Modélisation d'événements".

Liaisons

Au coeur de l'architecture orientée services se trouve le concept de *service*, une unité de fonctionnalité accomplie par une interaction entre les périphériques de traitement. Une *exportation* définit l'interface externe (ou le point d'accès) d'un module, de telle sorte que les composants SCA (Service Component Architecture) au sein du module puissent fournir leurs services à des clients externes. Une *importation* définit une interface vers des services situés à l'extérieur d'un module, de telle sorte que les services pussent être appelés depuis le module. Vous pouvez utiliser des *liaisons* spécifiques à un protocole avec des importations et des exportations pour spécifier les moyens de transport des données depuis ou vers le module.

Exportations

Les clients externes peuvent appeler des composants SCA dans un module d'intégration via une variété de protocoles (comme HTTP, JMS, MQ et RMI/IIOP) avec des données dans une variété de formats (comme XML, CSV, COBOL et JavaBeans). Les exportations sont des composants qui reçoivent ces requêtes de sources externes et appellent ensuite des composants IBM Business Process Manager à l'aide du modèle de programmation SCA.

Par exemple, dans la figure suivante, une exportation reçoit une requête via le protocole HTTP d'une application client. Les données sont transformées en un objet métier, le format utilisé par le composant

SCA. Le composant est ensuite appelé avec cet objet de données.

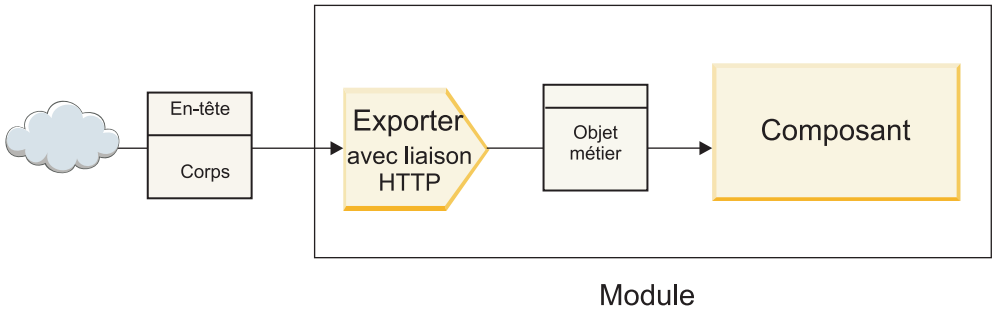


Figure 1. Exportation avec une liaison HTTP

Importations

Il est possible qu'un composant SCA souhaite appeler un service externe non SCA qui attend des données dans un format différent. Une importation est utilisée par le composant SCA pour appeler le service externe à l'aide du modèle de programmation SCA. L'importation appelle ensuite le service cible selon les attentes du service.

Par exemple, dans la figure suivante, une requête provenant d'un composant SCA est envoyée par l'importation vers un service externe. L'objet métier qui est au format utilisé par le composant SCA est transformé en format attendu par le service et le service est appelé.

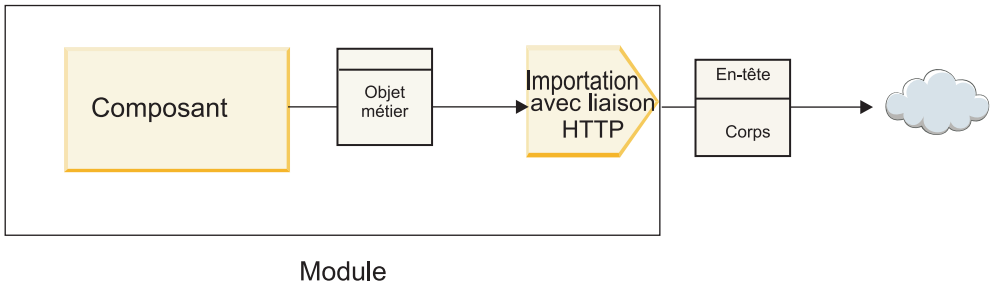


Figure 2. Importation avec une liaison HTTP

Liste de liaisons

Integration Designer permet de générer une liaison pour une importation ou une exportation et de configurer la liaison. Les types de liaisons disponibles sont décrites dans la liste suivante.

- SCA

  La liaison SCA qui est la liaison par défaut permet à votre service de communiquer avec des services dans d'autres modules SCA. Une importation avec une liaison SCA permet d'accéder à un service dans un autre module SCA. Une exportation avec une liaison SCA permet d'offrir un service à d'autres modules SCA.
- Service Web

  Une liaison de service Web vous permet d'accéder à un service externe par le biais de messages SOAP interopérables et des qualités de service. Vous pouvez également utiliser des liaisons de service Web pour inclure des pièces jointes dans le message SOAP.

  La liaison de service Web peut utiliser un protocole de transport de type SOAP/HTTP (SOAP via HTTP) ou SOAP/JMS (SOAP via JMS). Quel que soit le transport (HTTP ou JMS) utilisé pour transmettre les messages SOAP, les liaisons de service Web traitent toujours les interactions de demande/réponse de manière synchrone.

- HTTP

  La liaison HTTP vous permet d'accéder à un service externe via le protocole HTTP lorsque des messages non SOAP sont utilisés ou lorsque l'accès HTTP direct est requis. Cette liaison est utilisée lorsque vous utilisez des services Web qui sont basés sur le modèle HTTP (c'est-à-dire les services qui utilisent des opérations d'interface HTTP bien connues comme GET, PUT, DELETE, etc.).
- EJB (Enterprise JavaBeans)

  Les liaisons EJB permettent aux composants SCA d'interagir avec les services fournis par la logique métier Java EE exécutée sur un serveur Java EE.
- EIS

  Lorsqu'elle est utilisée avec un adaptateur de ressources JCA, la liaison EIS (Enterprise Information System) permet d'accéder à des services sur un système d'information d'entreprise ou de rendre vos services disponibles auprès de l'EIS.
- Liaisons JMS

  Les liaisons Java Message Service (JMS), JMS génériques et JMS WebSphere MQ (JMS MQ) sont utilisées pour des interactions avec des systèmes de messagerie où la communication asynchrone via les files d'attente de messages est critique pour la fiabilité.

  Une exportation avec une des liaisons JMS surveille l'arrivée d'un message dans une file d'attente et la réponse, le cas échéant, est envoyée de manière asynchrone à la file d'attente de réponses. Une importation avec une des liaisons JMS génère et envoie un message vers une file d'attente JMS et surveille, le cas échéant, l'arrivée d'une réponse dans une file d'attente.
  - JMS

    La liaison JMS vous permet d'accéder au fournisseur JMS imbriqué dans WebSphere.
  - JMS générique

    La liaison JMS générique vous permet d'accéder à un système de messagerie fournisseur non IBM.
  - JMS MQ

    La liaison JMS MQ vous permet d'accéder au sous-ensemble JMS d'un système de messagerie WebSphere MQ. Vous pourriez utiliser cette liaison lorsque le sous-ensemble JMS de fonctions est suffisant pour votre application.
- MQ

  La liaison WebSphere MQ vous permet de communiquer avec des applications natives MQ, en les insérant dans le cadre de l'architecture SOA et en fournissant un accès aux informations d'en-tête propres à MQ. Cette liaison est utile lorsque vous devez utiliser des fonctions natives MQ.

Présentation des liaisons d'importation et d'exportation

Une exportation vous permet de rendre les services d'un module d'intégration disponibles pour les clients externes et une importation permet à vos composants SCA au sein d'un module d'intégration d'appeler des services externes. La liaison associée à l'exportation ou à l'importation spécifie la relation entre les messages de protocole et les objets métier. Elle spécifie également la manière dont les opérations et les erreurs sont sélectionnées.

Flux d'informations via une exportation

Une exportation reçoit une requête destinée au composant auquel l'exportation est connectée via un transport spécifique déterminé par la liaison associée (par exemple, HTTP).

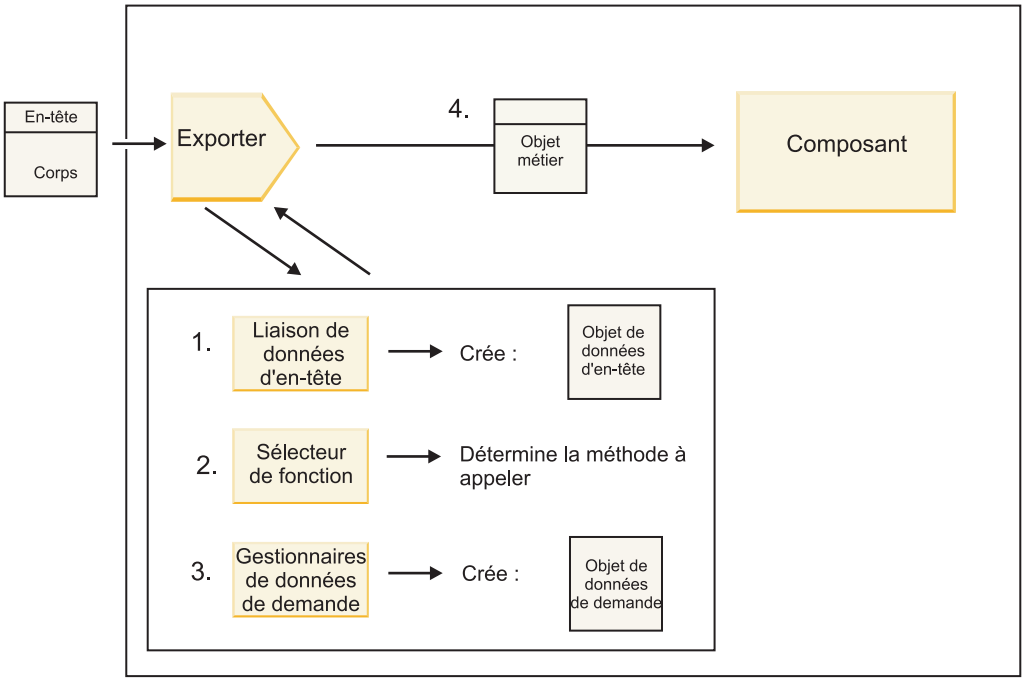


Figure 3. Flux d'une requête via l'exportation vers un composant

Lorsque l'exportation reçoit la requête, la séquence d'événements suivante se produit :

1. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête transforme l'en-tête de protocole en un objet de données d'en-tête.
2. Le sélecteur de fonction détermine le nom de la méthode native à partir du message de protocole. Le nom de la méthode native est mappée par la configuration d'exportation vers le nom d'une opération sur l'interface de l'exportation.
3. Le gestionnaire de données de requête ou la liaison de données sur la méthode transforme la requête en un objet métier de type requête.
4. L'exportation appelle la méthode de composant avec l'objet métier de type requête.
   - La liaison d'exportation HTTP, la liaison d'exportation des services Web, et la liaison d'exportation EJB appellent le composant SCA de manière synchrone.
   - Les liaisons d'exportation JMS, Generic JMS, MQ JMS et WebSphere MQ appellent le composant SCA de manière asynchrone.

Notez qu'une exportation peut propager les propriétés d'en-tête et utilisateur qu'elle reçoit via le protocole , si la propagation de contexte est activée. Les composants qui sont reliés à l'exportation peuvent ensuite accéder à ces propriétés d'en-têtes et utilisateur. Pour plus d'informations, consultez la rubrique «Propagation» du centre de documentation de WebSphere Integration Developer.

S'il s'agit d'une opération bidirectionnelle, le composant renvoie une réponse.

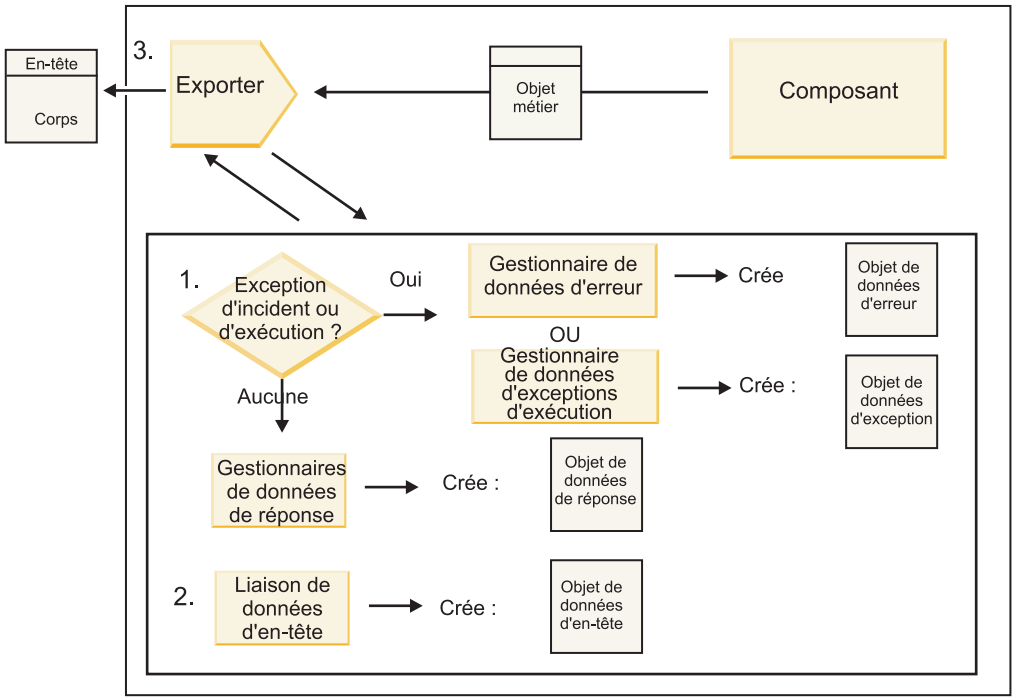


Figure 4. Flux d'une réponse en retour via l'exportation

La séquence d'étapes suivante se produit :

1. Si un message de réponse normal est reçu par la liaison d'exportation, le gestionnaire des données de réponse ou la liaison de données sur la méthode transforme l'objet métier en réponse.

   Si la réponse est une erreur, le gestionnaire des données d'erreur ou la liaison de données sur la méthode transforme l'erreur en une réponse d'erreur.

   Pour les liaisons d'exportation HTTP uniquement, si la réponse est une exception d'exécution, le gestionnaire des données d'exception d'exécution est appelé s'il est configuré.
2. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête transforme les objets de données d'en-tête en en-têtes de protocole.
3. L'exportation envoie la réponse du service via le transport.

Flux d'informations via une importation

Les composants envoient des requêtes aux services en dehors du module en utilisant une importation. La requête est envoyée via un transport spécifique déterminé par la liaison associée.

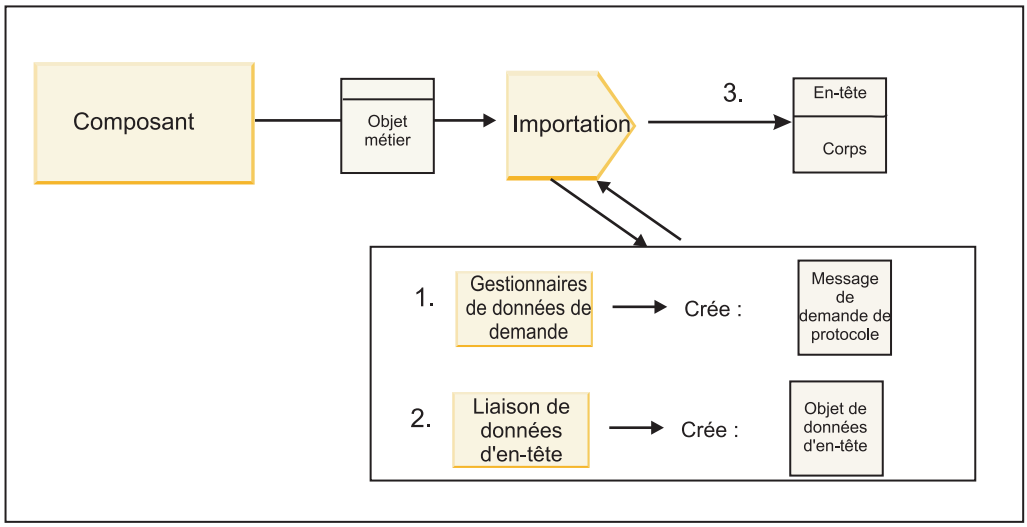


Figure 5. Flux d'un composant vers un service via l'importation

Le composant appelle l'importation avec un objet métier de type requête.

Remarque :

- La liaison d'importation HTTP, la liaison d'importation du service Web et la liaison d'importation EJB doivent être appelés de façon synchrone par le composant appelant.
- Les liaisons d'importation JMS, Generic JMS, MQ JMS et WebSphere MQ doivent être appelées de manière asynchrone.

Une fois que le composant a appelé l'importation, la séquence d'événements suivante se produit :

1. Le gestionnaire de données de requête ou la liaison de données sur la méthode transforme l'objet métier de type requête en un message de requête de protocole.
2. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête sur la méthode définit l'objet métier d'en-tête dans l'en-tête de protocole.
3. L'importation appelle le service avec la requête de service via le transport.

S'il s'agit d'une opération bidirectionnelle, le service renvoie une réponse et la séquence d'événements suivante se produit :

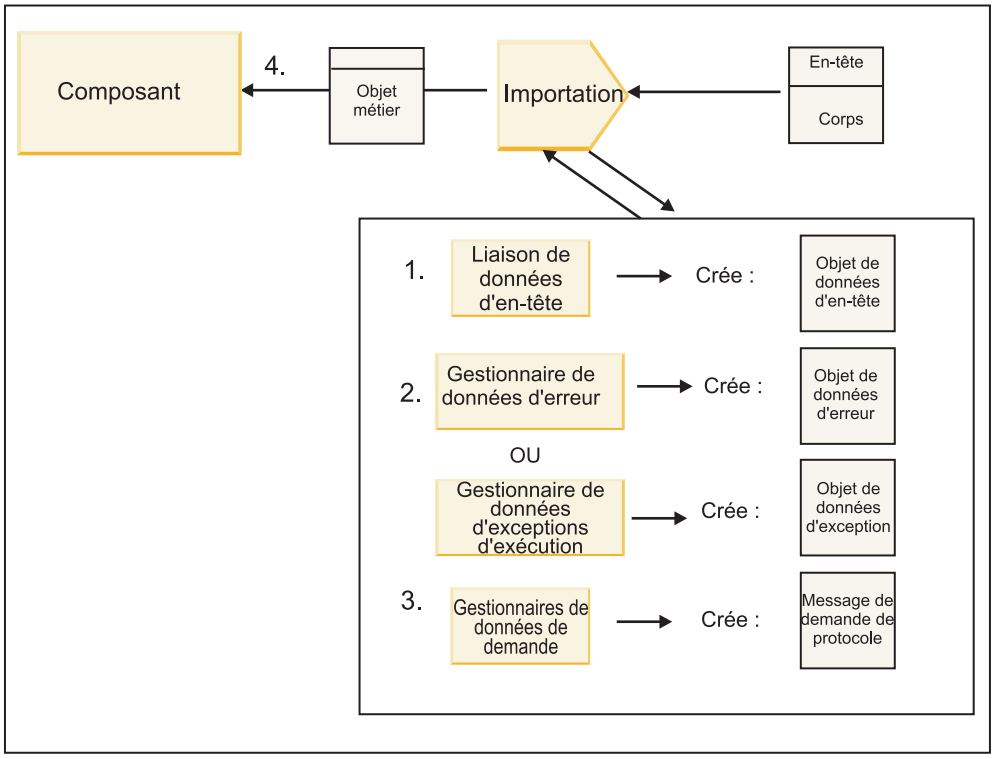


Figure 6. Flux d'une réponse en retour via l'importation

1. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête transforme l'en-tête de protocole en un objet de données d'en-tête.
2. Identification de la réponse afin de déterminer s'il s'agit d'une erreur.
   - Si la réponse est une erreur, le sélecteur d'erreurs inspecte l'erreur afin de déterminer vers quelle erreur WSDL elle mappe. Le gestionnaire des données d'erreur sur la méthode transforme ensuite l'erreur en une réponse d'erreur.
   - Si la réponse est une exception d'exécution, le gestionnaire des données d'exception d'exécution est appelé s'il est configuré.
3. Le gestionnaire de données de réponse ou la liaison sur la méthode transforme la réponse en un objet métier de réponse.
4. L'importation renvoie l'objet métier de réponse au composant.

Configuration des liaisons d'importation et d'exportation

Un des aspects clés des liaisons d'importation et d'exportation est constitué par la transformation du format des données qui indique la manière dont les données sont mappées (désérialisées) à partir d'un format de connexion natif vers un objet métier ou la manière dont elles sont mappées (sérialisées) depuis un objet métier vers un format de connexion natif. Pour les liaisons associées aux exportations, vous pouvez également spécifier un sélecteur de fonction pour identifier les opérations qui doivent être effectuées sur les données. Pour les liaisons associées aux exportations ou aux importations, vous pouvez indiquer le mode de gestion des échecs qui se sont produits lors du traitement.

En outre, vous pouvez spécifier des informations spécifiques au transport sur les liaisons. Par exemple, pour une liaison HTTP, vous pouvez spécifier l'adresse URL de noeud final. Pour la liaison HTTP, les informations spécifiques au transport sont décrites dans les rubriques «Génération d'une liaison d'importation HTTP» et «Génération d'une liaison d'exportation HTTP». Vous pouvez également trouver des informations sur les autres liaisons dans le centre de documentation.

Transformation du format des données dans les importations et exportations :

Lorsqu'une liaison d'exportation ou d'importation est configurée dans IBM Integration Designer, une des propriétés de configuration spécifiées concerne le format de données utilisé par la liaison.

- Pour les liaisons d'exportation, dans lesquelles une application client envoie des requêtes à un composant SCA et reçoit des réponses en retour, vous indiquez le format des données natives. En fonction du format, le système sélectionne le gestionnaire de données approprié ou la liaison de données pour transformer les données natives en un objet métier (qui est utilisé par le composant SCA) et inversement pour transformer l'objet métier en données natives (qui est la réponse à l'application client).
- Pour les liaisons d'importation, dans lesquelles un composant SCA envoie des requêtes à un service en dehors du module et reçoit des réponses en retour, vous indiquez le format de données des données natives. En fonction du format, le système sélectionne le gestionnaire de données approprié ou la liaison de données pour transformer l'objet métier en données natives et inversement.

IBM Business Process Manager fournit un ensemble de formats de données prédéfinis et de gestionnaires de données correspondants ou de liaisons de données qui prennent en charge les formats. Vous pouvez également créer vos propres gestionnaires de données personnalisés et enregistrer le format de données pour ces gestionnaires de données. Pour plus d'informations, voir «Développement des gestionnaires de données» dans le centre de documentation de IBM Integration Designer.

- Les *gestionnaires de données* ne dépendent pas d'un protocole et peuvent transformer les données d'un format à un autre. Dans IBM Business Process Manager, les gestionnaires de données transforment généralement des données natives (comme XML, CSV et COBOL) en un objet métier et inversement. Puisqu'ils ne dépendent pas d'un protocole, vous pouvez réutiliser le même gestionnaire de données avec toute une variété de liaisons d'exportation et d'importation. Par exemple, vous pouvez utiliser le même gestionnaire de données XML avec une liaison d'exportation ou d'importation HTTP ou avec une liaison d'exportation ou d'importation JMS.
- Les *liaisons de données* transforment également des données natives en objet métier (et inversement), mais elles sont spécifiques à un protocole. Par exemple, une liaison de données HTTP peut être utilisée uniquement avec une liaison d'exportation ou d'importation HTTP. Contrairement aux gestionnaires de données, une liaison de données HTTP ne peut pas être réutilisée avec une liaison d'exportation ou d'importation MQ.

  Remarque : Trois liaisons de données HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML et HTTPServiceGatewayDataBinding) sont dépréciées à partir de IBM Business Process Manager, version 7.0. Utilisez des gestionnaires de données chaque fois que possible.

Comme indiqué précédemment, vous pouvez créer des gestionnaires de données personnalisés, si nécessaire. Vous pouvez également créer des liaisons de données personnalisées. Toutefois, il est recommandé de créer des gestionnaires de données personnalisés car ils peuvent être utilisés avec plusieurs liaisons.

Gestionnaires de données :

Les gestionnaires de données sont configurés en fonction des liaisons d'importation et d'exportation pour transformer les données d'un format à un autre dans un environnement de protocole neutre. Plusieurs gestionnaires de données sont fournis avec le produit, mais vous pouvez également créer votre propre gestionnaire de données si nécessaire. Vous pouvez associer un gestionnaire de données à une liaison d'importation ou d'exportation à l'un des deux niveaux disponibles : vous pouvez l'associer à toutes les opérations dans l'interface de l'importation ou de l'exportation ou bien vous pouvez l'associer à une opération spécifique pour la requête ou la réponse.

Gestionnaires de données prédéfinis

IBM Integration Designer permet de spécifier le gestionnaire de données que vous voulez utiliser.

Le tableau suivant répertorie les gestionnaires de données prédéfinis et décrit également la manière dont chaque gestionnaire de données transforme les données entrantes et sortantes.

Remarque : Sauf indication contraire, ces gestionnaires de données peuvent être utilisés avec des liaisons JMS, Generic JMS, JMS MQ, WebSphere MQ et HTTP.
Voir la rubrique «Gestionnaires de données» dans le centre de documentation de Integration Designer pour obtenir des informations détaillées.

Tableau 23. Gestionnaires de données prédéfinis

Gestionnaire de données	Données natives vers objet métier	Objet métier vers données natives
ATOM	Analyse les flux ATOM dans un objet métier de flux ATOM.	Sérialise un objet métier de flux ATOM vers des flux ATOM.
Délimité	Analyse les données délimitées dans un objet métier.	Sérialise un objet métier vers des données délimitées, y compris CSV.
Largeur fixe	Analyse les données à largeur fixe dans un objet métier.	Sérialise un objet métier vers des données à largeur fixe.
Géré par WTX	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est dérivé par le gestionnaire de données.	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est dérivé par le gestionnaire de données.
Géré par le générateur d'appels WTX	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est fourni par l'utilisateur.	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est fourni par l'utilisateur.
JAXB	Sérialise des beans Java sous la forme d'objet métier à l'aide de règles de mappage définies par la spécification JAXB (Java Architecture for XML Binding).	Désérialise un objet métier Java à l'aide des règles de mappage définies par la spécification JAXB.
JAXWS **Remarque :** Le gestionnaire de données JAXWS ne peut être utilisé qu'avec la liaison EJB.	Utilisé par une liaison d'EJB pour transformer un objet Java de réponse ou un objet Java d'exception en objet métier de réponse à l'aide des règles de mappage définies par la spécification JAX-WS (Java API for XML Web Services).	Utilisé par une liaison d'EJB pour transformer un objet métier en paramètre de méthode Java à l'aide des règles de mappage définies par la spécification JAX-WS.
JSON	Analyse les données JSON dans un objet métier.	Sérialise un objet métier vers des données JSON.
Corps natif	Effectue l'analyse syntaxique des octets natifs (texte, mappe, flux ou objet) pour obtenir un des cinq objets métier de base (texte, octets, mappe, flux ou objet).	Transforme les cinq objets métier de base en octets, texte, mappe, flux ou objet.
SOAP	Analyse le message SOAP (et l'en-tête) dans un objet métier.	Sérialise un objet métier vers un message SOAP.
XML	Analyse les données XML dans un objet métier.	Sérialise un objet métier vers des données XML.
UTF8XMLDataHandler	Analyse les données XML codées en UTF-8 dans un objet métier.	Sérialise un objet métier dans des données XML codées en UTF-8 lors de l'envoi d'un message.

Création d'un gestionnaire de données

Pour plus d'informations sur la création d'un gestionnaire de données, consultez la rubrique «Développement des gestionnaires de données» du centre de documentation de Integration Designer.

Liaisons de données :

Les liaisons de données sont configurées en fonction des liaisons d'importation et d'exportation pour transformer les données d'un format à un autre. Les liaisons de données sont spécifiques à un protocole. Plusieurs liaisons de données sont fournies avec le produit, mais vous pouvez également créer votre propre liaison de données si nécessaire. Vous pouvez associer une liaison de données à une liaison d'importation ou d'exportation sur l'un des deux niveaux disponibles – vous pouvez l'associer avec toutes les opérations dans l'interface de l'importation ou de l'exportation ou bien vous pouvez l'associer à une opération spécifique pour la requête ou la réponse.

IBM Integration Designer vous permet de spécifier quelle liaison de données vous voulez utiliser ou de créer votre propre liaison de données. Une discussion traitant des liaisons de données est disponible dans la section «Présentation des liaisons JMS, JMS MQ et JMS génériques» du centre de documentation de IBM Integration Designer.

Liaisons JMS

Le tableau suivant répertorie les liaisons de données qui peuvent être utilisées :

- Liaisons JMS
- Liaisons JMS génériques
- Liaisons JMS WebSphere MQ

Le tableau comprend également une description des tâches que les liaisons de données effectuent.

Tableau 24. Liaisons de données prédéfinies pour les liaisons JMS

Liaison de données	Données natives vers objet métier	Objet métier vers données natives
Objet Java sérialisé	Transforme l'objet sérialisé Java en un objet métier (qui est mappé en tant que type d'entrée ou de sortie dans WSDL).	Sérialise un objet métier vers un objet sérialisé Java dans le message d'objet JMS.
Octets encapsulés	Extrait les octets du message d'octets JMS entrant et les encapsule dans l'objet métier JMSBytesBody.	Extrait les octets de l'objet métier JMSBytesBody et les encapsule dans le message d'octets JMS sortant
Entrée de mappe encapsulée	Extrait les informations de nom, de valeur et de type pour chaque entrée dans le message de mappe JMS entrant et crée une liste d'objets métier MapEntry. Encapsule ensuite la liste dans l'objet métier JMSMapBody	Extrait les informations de nom, de valeur et de type à partir de la liste MapEntry dans l'objet métier JMSMapBody business et crée les entrées correspondantes dans le message de mappe JMS sortant.
Objet encapsulé	Extrait l'objet du message d'objet JMS entrant et l'encapsule dans l'objet métier JMSObjectBody.	Extrait l'objet de l'objet métier JMSObjectBody et l'encapsule dans le message d'objet JMS sortant.
Texte encapsulé	Extrait le texte du message de texte JMS entrant et l'encapsule dans l'objet métier JMSTextBody.	Extrait le texte de l'objet métier JMSTextBody et l'encapsule dans le message de texte JMS sortant.

Liaisons WebSphere MQ

Le tableau suivant répertorie les liaisons de données qui peuvent être utilisées avec WebSphere MQ et décrit les tâches que les liaisons de données peuvent effectuer.

Tableau 25. Liaisons de données prédéfinies pour les liaisons WebSphere MQ

Liaison de données	Données natives vers objet métier	Objet métier vers données natives
Objet Java sérialisé	Transforme l'objet sérialisé Java à partir du message entrant en un objet métier (qui est mappé en tant que type d'entrée ou de sortie dans WSDL).	Transforme un objet métier en objet sérialisé Java dans le message sortant
Octets encapsulés	Extrait les octets du message d'octets MQ non structuré et les encapsule dans l'objet métier JMSBytesBody.	Extrait les octets d'un objet métier JMSBytesBody et encapsule ces octets dans le message d'octets MQ non structuré sortant.
Texte encapsulé	Extrait le texte du message de texte MQ non structuré et l'encapsule dans un objet métier JMSTextBody.	Extrait le texte d'un objet métier JMSTextBody et l'encapsule dans un message de texte MQ non structuré.
Entrée de flux encapsulée	Extrait les informations de nom et de type pour chaque entrée dans le message de flux JMS entrant et crée une liste d'objets métier StreamEntry. Encapsule ensuite la liste dans l'objet métier JMSStreamBody.	Extrait les informations de nom et de type à partir de la liste StreamEntry dans l'objet métier JMSStreamBody et crée les entrées correspondantes dans le message JMSStreamMessage sortant.

Outre les liaisons de données répertoriées dans le tableau 25, WebSphere MQ utilise également des liaisons de données d'en-tête. Pour plus d'informations, reportez-vous au centre de documentation de IBM Integration Designer.

Liaisons HTTP

Le tableau suivant répertorie les liaisons de données qui peuvent être utilisées avec HTTP et décrit les tâches que les liaisons de données peuvent effectuer.

Tableau 26. Liaisons de données prédéfinies pour les liaisons HTTP

Liaison de données	Données natives vers objet métier	Objet métier vers données natives
Octets encapsulés	Extrait les octets du corps du message HTTP entrant et les encapsule dans l'objet métier HTTPBytes.	Extrait les octets de l'objet métier HTTPBytes et les ajoute au corps du message HTTP sortant.
Texte encapsulé	Extrait le texte du corps du message HTTP entrant et l'encapsule dans l'objet métier HTTPText.	Extrait le texte de l'objet métier HTTPText et l'ajoute au corps du message HTTP sortant.

Sélecteurs de fonction dans les liaisons d'exportation :

Un sélecteur de fonction permet d'identifier les opérations qui doivent être effectuées sur les données pour un message de demande. Les sélecteurs de fonction sont configurés dans le cadre d'une liaison d'exportation.

Considérons une exportation SCA qui expose une interface. L'interface contient deux opérations : Créer et Mettre à jour. L'exportation a une liaison JMS qui lit à partir d'une file d'attente.

Lorsqu'un message arrive dans la file d'attente, l'exportation est transmise aux données associées, mais quelle opération de l'interface d'exportation doit être appelée sur le composant connecté ? L'opération est déterminée par le sélecteur de fonction et la configuration de la liaison d'exportation.

Le sélecteur de fonction renvoie le nom de la fonction native (le nom de la fonction dans le système client ayant envoyé le message). Le nom de la fonction native est ensuite mappé à l'opération ou au nom de la fonction sur l'interface associée à l'exportation. Par exemple, dans la figure suivante, le sélecteur de fonction renvoie le nom de la fonction native (CRT) à partir du message entrant, le nom de la fonction native est mappé à l'opération Créer et l'objet métier est envoyé au composant SCA avec l'opération Créer.

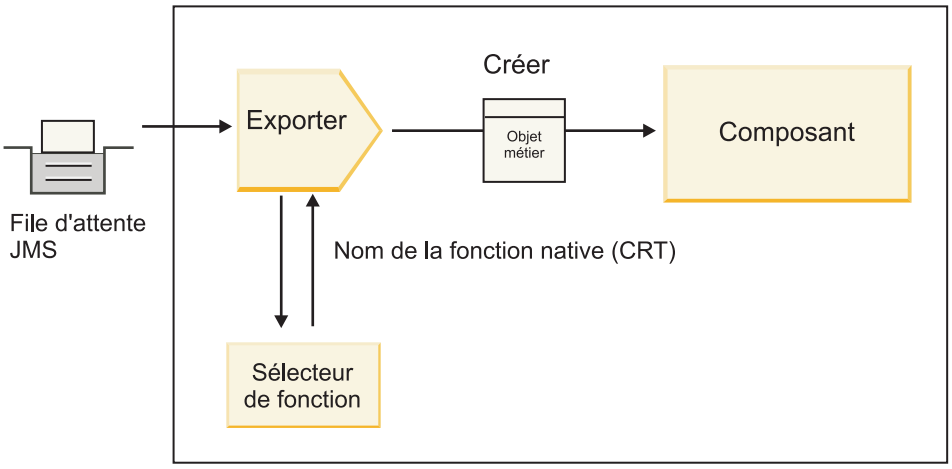


Figure 7. Sélecteur de fonction

Si l'interface ne présente qu'une opération, il n'est pas nécessaire de spécifier un sélecteur de fonction.

Plu sieurs sélecteurs de fonction préintégrés sont disponibles et répertoriés dans les sections ci-dessous.

Liaisons JMS

Le tableau suivant répertorie les sélecteurs de fonction qui peuvent être utilisés avec :

- Liaisons JMS
- Liaisons JMS génériques
- Liaisons JMS WebSphere MQ

Tableau 27. Sélecteurs de fonction prédéfinis pour les liaisons JMS

Sélecteur de fonction	Description
Sélecteur de fonction JMS pour des liaisons de données JMS simples	Utilise la propriété JMSType du message pour sélectionner le nom de l'opération.
Sélecteur de fonction de propriété d'en-tête JMS	Renvoie la valeur de la propriété de chaîne JMS, TargetFunctionName, à partir de l'en-tête.
Sélecteur de fonction de passerelle de service JMS	Détermine si la requête est une opération unidirectionnelle ou bidirectionnelle en examinant l'ensemble de propriétés JMSReplyTo par le client.

Liaisons WebSphere MQ

Le tableau suivant répertorie les sélecteurs de fonction qui peuvent être utilisés avec les liaisonsWebSphere MQ.

Tableau 28. Sélecteurs de fonction prédéfinis pour les liaisons WebSphere MQ

Sélecteur de fonction	Description
Sélecteur de fonction MQ handleMessage	Renvoie un message handleMessage comme valeur qui est mappé à l'aide des liaisons de méthode d'exportation au nom d'une opération sur l'interface.
MQ utilise le sélecteur de fonction JMS par défaut	Lit l'opération native à partir de la propriété TargetFunctionName du dossier d'un en-tête MQRFH2.
MQ utilise le format du corps de message comme fonction native	Recherche la zone Format du dernier en-tête et renvoie cette zone sous forme de chaîne.
Sélecteur de fonction de type MQ	Crée une méthode dans votre liaison d'exportation par extraction de l'adresse URL contenant les propriétés Msd, Set, Type et Format de l'en-tête MQRFH2.
Sélecteur de fonction de passerelle de service MQ	Utilise la propriété MsgType de l'en-tête MQMD pour déterminer le nom de l'opération.

Liaisons HTTP

Le tableau suivant répertorie les sélecteurs de fonction qui peuvent être utilisés avec les liaisons HTTP.

Tableau 29. Sélecteurs de fonction prédéfinis pour les liaisons HTTP

Sélecteur de fonction	Description
Sélecteur de fonction HTTP basé sur l'en-tête TargetFunctionName	Utilise la propriété de l'en-tête HTTP TargetFunctionName du client pour déterminer l'opération à appeler au moment de l'exécution à partir de l'exportation.
Sélecteur de fonction HTTP basé sur l'adresse URL et la méthode HTTP	Utilise le chemin relatif à partir de l'adresse URL ajoutée avec la méthode HTTP à partir du client pour déterminer l'opération native définie sur l'exportation.
Sélecteur de fonction de passerelle de service HTTP basé sur l'adresse URL avec un nom d'opération	Détermine la méthode à appeler en fonction de l'adresse URL si "operationMode = oneway" a été ajouté à l'URL de demande.

Remarque : Vous pouvez également créer votre propre sélecteur de fonction à l'aide d'IBM Integration Designer. Les informations de création d'un sélecteur de fonction se trouvent dans le centre de documentation d'IBM Integration Designer. Par exemple, une description de la création d'un sélecteur de fonction pour les liaisons WebSphere MQ est disponible dans la rubrique «Présentation des sélecteurs de fonction MQ».

Gestion des erreurs :

Vous pouvez configurer vos liaisons d'importation et d'exportation pour gérer les erreurs (par exemple, les exceptions métier) qui se produisent lors du traitement en spécifiant des gestionnaires de données d'erreur. Vous pouvez configurer un gestionnaire de données d'erreur sur trois niveaux : vous pouvez associer un gestionnaire de données d'erreur à une erreur, à une opération ou pour toutes les opérations à une liaison.

Un gestionnaire de données d'erreur traite les données d'erreur et les convertit au format correct pour les envoyer via la liaison d'importation ou d'exportation.

- Pour une liaison d'exportation, le gestionnaire de données d'erreur transforme l'objet métier d'exception envoyé depuis le composant en un message de réponse qui peut être utilisé par l'application client.

- Pour une liaison d'importation, le gestionnaire de données d'erreur transforme les données d'erreur ou le message de réponse provenant d'un service en un objet métier d'exception qui peut être utilisé par le composant SCA.

Pour les liaisons d'importation, la liaison appelle le sélecteur d'erreurs qui détermine si le message de réponse est une réponse normale, une erreur métier ou une exception d'exécution.

Vous pouvez spécifier un gestionnaire de données d'erreur pour une erreur, une opération particulière et pour toutes les opérations avec une liaison.

- Si le gestionnaire de données d'erreur est défini sur les trois niveaux, le gestionnaire de données associé à une erreur particulière est appelé.
- Si les gestionnaires de données d'erreur sont définis aux niveaux de l'opération et de la liaison, le gestionnaire de données associé à l'opération est appelé.

Deux éditeurs sont utilisés dans IBM Integration Designer pour spécifier la gestion des erreurs. L'éditeur d'interface est utilisé pour indiquer s'il y aura une erreur sur une opération. Après avoir généré une liaison avec cette interface, l'éditeur de la vue des propriétés vous permet de configurer la manière dont l'erreur sera gérée. Pour plus d'informations, voir la rubrique «Sélecteur d'erreur» dans le centre de documentation de IBM Integration Designer.

Gestion des erreurs dans les liaisons d'exportation :

Lorsqu'une erreur se produit lors du traitement de la requête d'une application client, la liaison d'exportation peut renvoyer les informations d'erreur au client. Vous pouvez configurer la liaison d'exportation pour spécifier la manière dont l'erreur doit être traitée et renvoyée au client.

Vous configurez la liaison d'exportation à l'aide de IBM Integration Designer.

Lors du traitement de la requête, un client appelle une exportation avec une requête et l'exportation appelle le composant SCA. Lors du traitement de la requête, le composant SCA peut renvoyer une réponse métier ou émettre une exception métier de service ou une exception d'exécution de service. Lorsque cela se produit, la liaison d'exportation transforme l'exception en un message d'erreur et l'envoie au client, comme indiqué dans la figure ci-dessous et décrit dans les sections suivantes.

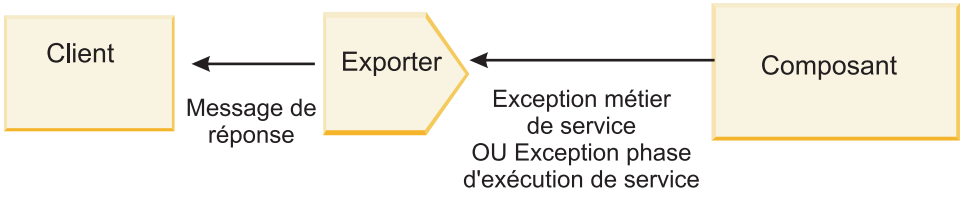


Figure 8. Manière dont les informations sont envoyées du composant vers le client via la liaison d'exportation

Vous pouvez créer un gestionnaire de données personnalisé ou une liaison de données pour gérer les erreurs.

Erreurs métier

Les erreurs métier sont des erreurs ou des exceptions métier qui se produisent lors du traitement.

Considérez l'interface suivante présentant une opération createCustomer. Cette opération se caractérise par deux erreurs métier définies : CustomerAlreadyExists et MissingCustomerId.

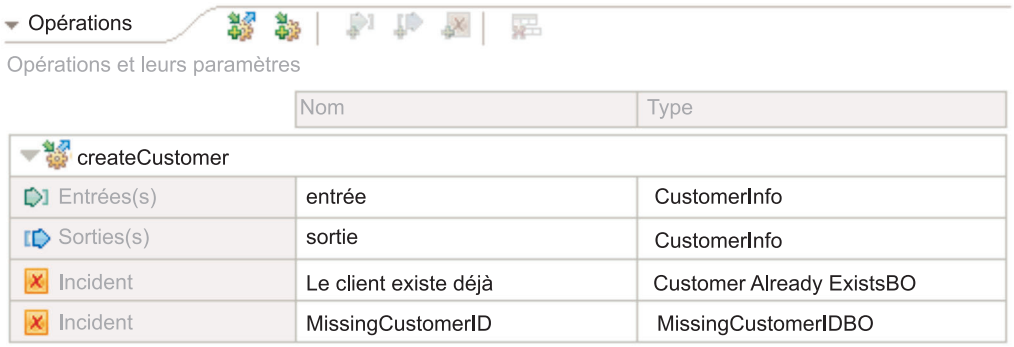


Figure 9. Interface avec deux erreurs

Dans cet exemple, si un client envoie une requête pour créer un client (à ce composant SCA) et que le client existe déjà, le composant émet une erreur CustomerAlreadyExists vers l'exportation. L'exportation doit propager cette erreur métier en retour au client appelant. Pour ce faire, elle utilise le gestionnaire de données configuré sur la liaison d'exportation.

Lorsqu'une erreur métier est reçue par la liaison d'exportation, le traitement suivant se produit :

1. La liaison détermine le gestionnaire de données d'erreurs à appeler pour gérer les erreurs. Si l'exception métier de service contient le nom de l'erreur, le gestionnaire de données configuré sur l'erreur est appelé. Si l'exception métier de service ne contient pas le nom de l'erreur, ce nom est dérivé en faisant correspondre les types d'erreur.
2. La liaison appelle le gestionnaire de données d'erreurs avec l'objet de données à partir de l'exception métier de service.
3. Le gestionnaire de données d'erreur transforme l'objet données d'erreur en un message de réponse et le renvoie à la liaison d'exportation.
4. L'exportation renvoie le message de réponse au client.

Si l'exception métier de service contient le nom de l'erreur, le gestionnaire de données configuré sur l'erreur est appelé. Si l'exception métier de service ne contient pas le nom de l'erreur, ce nom est dérivé en faisant correspondre les types d'erreur.

Exceptions d'exécution

Une exception d'exécution est une exception qui se produit dans l'application SCA lors du traitement d'une requête qui ne correspond pas à une erreur métier. Contrairement aux erreurs métier, les exceptions d'exécution ne sont pas définies sur l'interface.

Dans certains scénarios, il est possible que vous souhaitiez propager ces exceptions d'exécution à l'application client de telle sorte que l'application client puisse entreprendre l'action appropriée.

Par exemple, si un client envoie une requête (au composant SCA) pour créer un client et qu'une erreur d'autorisation se produit lors du traitement de la requête, le composant émet une exception d'exécution. Cette exception d'exécution doit être propagée en retour au client appelant de telle sorte qu'il puisse entreprendre l'action appropriée en ce qui concerne l'autorisation. Cette opération est accomplie par le gestionnaire de données d'exception d'exécution configuré sur la liaison d'exportation.

Remarque : Vous pouvez configurer un gestionnaire de données d'exception d'exécution uniquement sur des liaisons HTTP.

Le traitement d'une exception d'exécution est identique au traitement d'une erreur métier. Si un gestionnaire de données d'exception d'exécution était configuré, le traitement suivant se produit :

1. La liaison d'exportation appelle le gestionnaire de données approprié avec l'exception d'exécution de service.
2. Le gestionnaire de données transforme l'objet données d'erreur en un message de réponse et le renvoie à la liaison d'exportation.
3. L'exportation renvoie le message de réponse au client.

La gestion des erreurs et la gestion des exceptions d'exécution sont facultatives. Si vous ne voulez pas propager les erreurs et les exceptions d'exécution au client appelant, ne configurez pas le gestionnaire de données d'erreur ou le gestionnaire de données d'exception d'exécution.

Gestion des erreurs dans les liaisons d'importation :

Un composant utilise une importation pour envoyer une requête à un service en dehors du module. Lorsqu'une erreur se produit lors du traitement de la requête, le service renvoie l'erreur à la liaison d'importation. Vous pouvez configurer la liaison d'importation pour spécifier la manière dont l'erreur doit être traitée et renvoyée au composant.

Vous configurez la liaison d'importation à l'aide de IBM Integration Designer. Vous pouvez spécifier un gestionnaire de données d'erreur (ou une liaison de données) et vous pouvez également spécifier un sélecteur d'erreurs.

Gestionnaires de données d'erreur

Le service qui traite la requête envoie des informations d'erreur à la liaison d'importation sous la forme d'une exception ou un message de réponse contenant les données d'erreur.

La liaison d'importation transforme l'exception de service ou le message de réponse en une exception métier de service ou une exception d'exécution de service, comme indiqué dans la figure ci-dessous et décrit dans les sections suivantes.

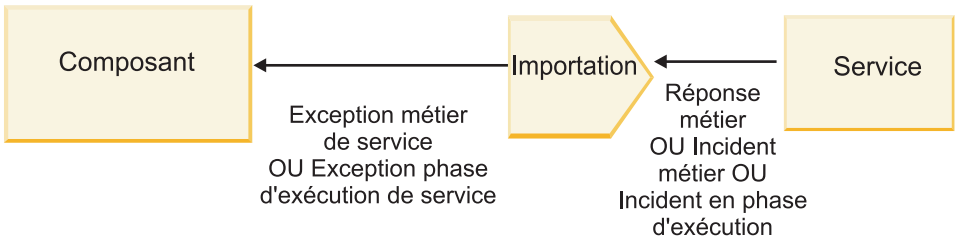


Figure 10. Manière dont les informations d'erreur sont envoyées depuis le service au composant via l'importation

Vous pouvez créer un gestionnaire de données personnalisé ou une liaison de données pour gérer les erreurs.

Sélecteurs d'erreurs

Lorsque vous configurez une liaison d'importation, vous pouvez spécifier un sélecteur d'erreurs. Le sélecteur d'erreurs détermine si la réponse d'importation est une réponse réelle, une exception métier ou une erreur d'exécution. Il détermine également, à partir de l'en-tête ou du corps de la réponse, le nom de l'erreur native qui est mappé par la configuration de liaison au nom d'une erreur dans l'interface associée.

Deux types de sélecteurs d'erreurs préintégrés sont disponibles avec les importations JMS, JMS MQ, JMS génériques, WebSphere MQ et HTTP :

Tableau 30. Sélecteurs d'erreurs préintégrés

Type de sélecteur d'erreurs	Description
Basé sur les en-têtes	Indique si un message de réponse est une erreur métier, une exception d'exécution ou un message normal en fonction des en-têtes dans le message de réponse.
SOAP	Détermine si le message de réponse SOAP est une réponse normale, une erreur métier ou une exception d'exécution.

Les exemples ci-dessous illustrent des sélecteurs d'erreurs basés sur les en-têtes et le sélecteur d'erreurs SOAP.

- Sélecteur d'erreurs basé sur les en-têtes

  Si une application veut indiquer que le message entrant est une erreur métier, il doit y avoir deux en-têtes dans le message entrant pour les erreurs métier, comme indiqué ci-après :

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <user defined native fault name>
  ```

  Si une application veut indiquer que le message de réponse entrant est une exception d'exécution, il doit y avoir un en-tête dans le message entrant comme indiqué ci-après :

  ```
  Header name = FaultType, Header value = Runtime
  ```

- Sélecteur d'erreurs SOAP

  Une erreur métier peut être envoyée dans le cadre d'un message SOAP avec l'en-tête SOAP personnalisé suivant : "CustomerAlreadyExists" est le nom de l'erreur dans ce cas.

  ```
  <ibmSoap:BusinessFaultName
  xmlns:ibmSoap="http://www.ibm.com/soap">CustomerAlreadyExists
  <ibmSoap:BusinessFaultName>
  ```

Le sélecteur d'erreurs est facultatif. Si vous ne spécifiez aucun sélecteur d'erreurs, la liaison d'importation ne peut pas déterminer le type de réponse. En conséquence, la liaison la traite comme une réponse métier et appelle le gestionnaire de données de réponse ou la liaison de données.

Vous pouvez créer un sélecteur d'erreurs personnalisé. Les étapes permettant de créer un sélecteur d'erreurs personnalisé se trouvent dans la rubrique «Developing a custom fault selector» du centre de documentation IBM Integration Designer.

Erreurs métier

Une erreur métier peut se produire lorsqu'il existe une erreur dans le traitement d'une requête. Par exemple, si vous envoyez une requête pour créer un client et que le client existe déjà, le service envoie une exception métier à la liaison d'importation.

Lorsqu'une exception métier est reçue par la liaison, les étapes du traitement dépendent de la configuration d'un sélecteur d'erreurs pour la liaison.

- Si aucun sélecteur d'erreurs n'a été configuré, la liaison appelle le gestionnaire de données de réponse ou la liaison de données.
- Si un sélecteur d'erreurs a été configuré, le traitement suivant se produit :
  1. La liaison d'importation appelle le sélecteur d'erreurs pour déterminer si la réponse est une erreur métier, une réponse métier ou une erreur d'exécution.
  2. Si la réponse est une erreur métier, la liaison d'importation appelle le sélecteur d'erreurs pour fournir le nom de l'erreur native.
  3. La liaison d'importation identifie l'erreur WSDL correspondant au nom de l'erreur native renvoyée par le sélecteur d'erreurs.

4. La liaison d'importation détermine le gestionnaire de données d'erreur configuré pour cette erreur WSDL.
5. La liaison d'importation appelle ce gestionnaire de données d'erreur avec les données d'erreur.
6. Le gestionnaire de données d'erreur transforme les données d'erreur en un objet de données et le renvoie à la liaison d'importation.
7. La liaison d'importation construit un objet d'exception métier de service avec l'objet de données et le nom de l'erreur.
8. L'importation renvoie l'objet d'exception métier de service au composant.

Exceptions d'exécution

Une exception d'exécution peut se produire lorsqu'il existe un problème de communication avec le service. Le traitement d'une exception d'exécution est identique au traitement d'une exception métier. Si un sélecteur d'erreurs a été configuré, le traitement suivant se produit :

1. La liaison d'importation appelle le gestionnaire de données d'exception d'exécution approprié avec les données d'exception.
2. Le gestionnaire de données d'exception d'exécution transforme les données d'exception en un objet d'exception d'exécution de service et le renvoie à la liaison d'importation.
3. L'importation renvoie l'objet d'exception d'exécution de service au composant.

Interopérabilité entre les modules SCA et les services Open SCA

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) offre un modèle de programmation simple, mais puissant, permettant de créer des applications basées sur les spécifications Open SCA. Les modules SCA de IBM Business Process Manager utilisent des liaisons d'importation et d'exportation pour interagir avec les services Open SCA développés dans un environnement Rational Application Developer et hébergés par WebSphere Application Server Feature Pack for Service Component Architecture.

Une application SCA appelle une application Open SCA par l'intermédiaire d'une liaison d'importation. Une application SCA reçoit un appel d'une application Open SCA par l'intermédiaire d'une liaison d'exportation. Une liste des liaisons prises en charge est présentée dans «Appel de services sur des liaisons interopérables», à la page 70.

Appel de services Open SCA à partir de modules SCA

Les applications SCA développées avec IBM Integration Designer peuvent appeler des applications Open SCA développées dans un environnement Rational Application Developer. Cette section illustre un exemple d'appel d'un service Open SCA à partir d'un module SCA à l'aide d'une liaison d'importation SCA.

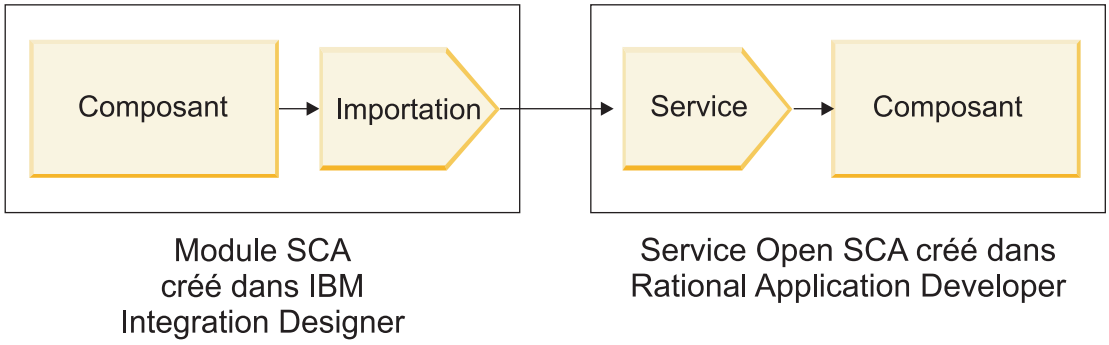


Figure 11. Composant de module SCA appelant un service Open SCA

Aucune configuration spéciale n'est requise pour appeler un service Open SCA.

Pour vous connecter à un service Open SCA par l'intermédiaire d'une liaison d'importation SCA, spécifiez le nom de composant et le nom de service du service Open SCA dans la liaison d'importation.

1. Pour obtenir le nom du composant cible et du service à partir du composite Open SCA, procédez comme suit :
   a. Vérifiez que la page **Propriétés** est ouverte en cliquant sur **Fenêtre** > **Afficher la vue** > **Propriétés**.
   b. Ouvrez l'éditeur de composite en cliquant deux fois sur le diagramme de composite qui contient le composant et le service. Par exemple, pour un composant intitulé **customer**, le diagramme du composite est **customer.composite_diagram**.
   c. Cliquez sur le composant cible.
   d. Dans la zone **Nom** de la page **Propriétés**, relevez le nom du composant cible.
   e. Cliquez sur l'icône de service associé au composant.
   f. Dans la zone **Nom** de la page **Propriétés**, relevez le nom du service.
2. Pour configurer l'importation IBM Business Process Manager afin de la connecter au service Open SCA, procédez comme suit :
   a. Dans IBM Integration Designer, accédez à l'onglet **Propriétés** de l'importation SCA à connecter au service Open SCA.
   b. Dans la zone **Nom du module**, entrez le nom de composant de l'étape 1d.
   c. Dans la zone **Nom d'exportation**, entrez le nom de service de l'étape 1f.
   d. Sauvegardez votre travail en appuyant sur les touches Ctrl+S.

Appel de modules SCA à partir de services Open SCA

Les applications Open SCA développées dans un environnement Rational Application Developer peuvent appeler des applications SCA développées avec IBM Integration Designer. Cette section illustre un exemple d'appel d'un module SCA (par l'intermédiaire d'une liaison d'exportation SCA) à partir d'un service Open SCA.

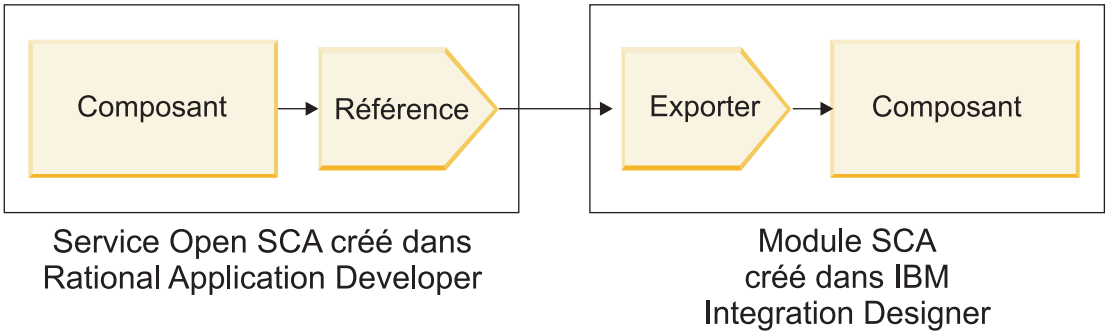


Figure 12. Service Open SCA appelant un composant dans un module SCA

Pour vous connecter à un composant SCA par l'intermédiaire d'une liaison de référence Open SCA, vous spécifiez le nom du module et le nom de l'exportation.

1. Pour obtenir le nom du composant cible et de l'exportation, procédez comme suit :
   a. Dans IBM Integration Designer, ouvrez le module dans l'éditeur d'assemblage en cliquant deux fois sur le module.
   b. Cliquez sur l'exportation.
   c. Dans la zone **Nom** de la page **Propriétés**, relevez le nom de l'exportation.
2. Configurez la référence Open SCA à connecter au module IBM Business Process Manager et à l'exportation :
   a. Dans Rational Application Developer, ouvrez l'éditeur de composite en cliquant deux fois sur le diagramme de composite qui contient le composant et le service.
   b. Cliquez sur l'icône de référence de la référence de composant pour en afficher les propriétés dans la page **Propriétés**.
   c. Cliquez sur l'onglet **Liaison** dans la partie gauche de la page.
   d. Cliquez sur **Liaisons**, puis sur **Ajouter**.
   e. Sélectionnez la liaison **SCA**.
   f. Dans la zone **Uri**, entrez le nom du module IBM Business Process Manager, suivi d'une barre oblique («/»), suivie du nom de l'exportation (que vous avez déterminé à l'étape 1c).
   g. Cliquez sur **OK**.
   h. Sauvegardez votre travail en appuyant sur les touches Ctrl+S.

Appel de services sur des liaisons interopérables

Les liaisons ci-après sont prises en charge pour l'interopérabilité avec un service Open SCA.

- Liaison SCA

  Dans IBM Business Process Manager, lorsqu'un module SCA appelle un service Open SCA par l'intermédiaire d'une liaison d'importation SCA, les styles d'appel suivants sont pris en charge :

  – Asynchrone (unidirectionnel)
  – Synchrone (demande/réponse)

  L'interface d'importation SCA et l'interface de service Open SCA doivent utiliser une interface WSDL compatible avec l'interopérabilité des services Web (WS-I).

  Notez que la liaison SCA prend en charge la propagation des transactions et des contextes de sécurité.
- Liaison de service Web (JAX-WS) avec protocole SOAP1.1/HTTP ou SOAP1.2/HTTP

  L'interface d'importation SCA et l'interface de service Open SCA doivent utiliser une interface WSDL compatible avec l'interopérabilité des services Web (WS-I).

  En outre, les qualités de service suivantes sont prises en charge :

  - Transaction atomique des services Web
  - Sécurité des services Web
- Liaison EJB

  Une interface Java est utilisée pour définir l'interaction entre un module SCA et un service Open SCA lorsque la liaison EJB est utilisée.

  Notez que la liaison EJB prend en charge la propagation des transactions et des contextes de sécurité.
- Liaisons JMS

  L'interface d'importation SCA et l'interface de service Open SCA doivent utiliser une interface WSDL compatible avec l'interopérabilité des services Web (WS-I).

  Les fournisseurs JMS pris en charge sont les suivants :
  - WebSphere Platform Messaging (Liaison JMS)
  - WebSphere MQ (Liaison JMS MQ)

Remarque : Les graphiques métier ne sont pas interopérables entre les liaisons SCA et ne sont donc pas prises en charge dans les interfaces utilisées pour interagir avec WebSphere Application Server Feature Pack for Service Component Architecture.

Types de liaison

Vous pouvez utiliser des *liaisons* spécifiques à un protocole avec des importations et des exportations pour spécifier les moyens de transport des données depuis ou vers un module.

Sélection des liaisons appropriées :

Lorsque vous créez une application, vous devez savoir comment sélectionner la liaison la plus appropriée aux besoins de votre application.

Les liaisons disponibles dans IBM Integration Designer offrent une variété de choix. Cette liste permet d'identifier le type de liaison le plus approprié pour les besoins de votre application.

Choisissez une liaison *SCA (Service Component Architecture)* lorsque les facteurs suivants sont applicables :
- Tous les services sont contenus dans des modules, c'est-à-dire qu'il n'existe pas de services externes.
- Vous souhaitez séparer les fonctions dans des modules SCA différents qui interagissent directement entre eux.
- Les modules sont étroitement couplés.

Choisissez une liaison *service Web* lorsque ces facteurs sont applicables :
- Vous devez accéder à un service externe via Internet ou fournir un service via Internet.
- Les services sont faiblement couplés.
- Vous préférez les communications synchrones ; une demande d'un service peut attendre une réponse d'un autre service.
- Le protocole du service externe auquel vous accédez ou du service à fournir est SOAP/HTTP ou SOAP/JMS.

Choisissez une liaison *HTTP* si les facteurs suivants sont applicables :
- Vous devez accéder à un service externe via Internet ou fournir un service par ce même biais et vous utilisez d'autres services Web tels que GET, PUT et DELETE.
- Les services sont faiblement couplés.
- Vous préférez les communications synchrones ; une demande d'un service peut attendre une réponse d'un autre service.

Choisissez une liaison *EJB (Enterprise Java Beans)* si les facteurs suivants sont applicables :

- La liaison est destinée à un service importé qui est lui-même un EJB ou auquel les clients EJB doivent accéder.
- Le service importé est faiblement couplé.
- Les interactions EJB avec état ne sont pas requises.
- Vous préférez les communications synchrones ; une demande d'un service peut attendre une réponse d'un autre service.

Choisissez une liaison *EIS (Enterprise Information Systems)* si les facteurs suivants sont applicables :
- Vous devez accéder à un service sur un système EIS à l'aide d'un adaptateur de ressources.
- Vous préférez une transmission de données synchrone à une transmission asynchrone.

Choisissez une liaison *JMS (Java Message Service)* lorsque les facteurs suivants sont applicables :

Important : Il existe plusieurs types de liaison JMS. Si vous prévoyez d'échanger des messages SOAP à l'aide de JMS, choisissez la liaison de service Web avec le protocole SOAP/JMS. Voir «Liaisons de service Web», à la page 73.
- Vous devez accéder à un système de messagerie.
- Les services sont faiblement couplés.
- Vous préférez une transmission de données asynchrone à une transmission synchrone.

Choisissez une liaison *JMS (Generic Java Message Service)* lorsque les facteurs suivants sont applicables :
- Vous devez accéder à un système de messagerie d'un fournisseur non IBM.
- Les services sont faiblement couplés.
- La fiabilité est plus importante que les performances ; vous préférez une transmission de données asynchrone à une transmission synchrone.

Choisissez une liaison *MQ (Message Queue)* si les facteurs suivants sont applicables :
- Vous devez accéder à un système de messagerie WebSphere MQ et utiliser les fonctions natives de MQ.
- Les services sont faiblement couplés.
- La fiabilité est plus importante que les performances ; vous préférez une transmission de données asynchrone à une transmission synchrone.

Choisissez une liaison *JMS MQ* si les facteurs suivants sont applicables :
- Vous devez accéder à un système de messagerie WebSphere MQ, mais pouvez le faire dans un contexte JMS ; le sous-ensemble JMS de fonctions est suffisant pour votre application.
- Les services sont faiblement couplés.
- La fiabilité est plus importante que les performances ; vous préférez une transmission de données asynchrone à une transmission synchrone.

Liaisons SCA :

Une liaison SCA (Service Component Architecture) permet à un service de communiquer avec d'autres services dans d'autres modules. Une importation avec une liaison SCA permet d'accéder à un service dans un autre module SCA. Une exportation avec une liaison SCA permet d'offrir un service à d'autres modules.

Utilisez IBM Integration Designer pour générer et configurer des liaisons de service Web pour les importations et exportations dans des modules SCA.

Si des modules sont exécutés sur le même serveur ou déployés dans le même cluster, une liaison SCA représente la liaison la plus simple et la plus rapide à utiliser.

Une fois que le module contenant la liaison SCA est déployé sur le serveur, vous pouvez utiliser la console d'administration pour afficher des informations sur la liaison ou, dans le cas d'une liaison d'importation, pour changer les propriétés sélectionnées de cette liaison.

Liaisons de service Web :

Une liaison de service Web représente le moyen par lequel les messages sont transmis d'un composant SCA (Service Component Architecture) vers un service Web (et inversement).

Présentation des liaisons de service Web : 

Une liaison d'importation de service Web vous permet d'appeler un service Web externe depuis vos composants SCA (Service Component Architecture). Une liaison d'exportation de service Web vous permet d'exposer vos composants SCA aux clients en tant que services Web.

Avec une liaison de service Web, vous pouvez accéder à des services externes via des messages SOAP interopérables et des qualités de service (QoS).

Utilisez Integration Designer pour générer et configurer des liaisons de service Web pour les importations et exportations dans des modules SCA. Les types suivants de liaisons de service Web sont disponibles :

- SOAP1.2/HTTP et SOAP1.1/HTTP

  Ces liaisons se basent sur JAX-WS (Java API for XML Web Services), une API de programmation Java permettant de créer des services Web.

  – Utilisez SOAP1.2/HTTP si votre service Web respecte la spécification SOAP 1.2.
  – Utilisez SOAP1.1/HTTP si votre service Web respecte la spécification SOAP 1.1.

  Important : Lorsque vous déployez une application avec une liaison de service Web (JAX-WS), vérifiez que l'option **Démarrer les composants en fonction des besoins** n'est pas sélectionnée dans le serveur cible. Pour plus de détails, voir «Vérification de la configuration du serveur», à la page 82.

  Lorsque vous sélectionnez l'une de ces liaisons, vous pouvez envoyer des pièces jointes avec vos messages SOAP.

  Les liaisons de service Web sont compatibles avec les messages SOAP standard. Toutefois, en utilisant l'une des liaisons JAX-WS de service Web, vous pouvez personnaliser la manière dont les messages SOAP sont analysés ou écrits. Par exemple, vous pouvez gérer les éléments non standard dans des messages SOAP ou appliquer un traitement supplémentaire au message SOAP. Lorsque vous configurez la liaison, vous spécifiez un gestionnaire de données personnalisé qui effectue ce traitement sur le message SOAP.

  Vous pouvez utiliser des ensembles de règles avec une liaison de service Web (JAX-WS). Un ensemble de règles regroupe différents types de règles qui procurent chacun une qualité de service définie (QoS). Par exemple, l'ensemble de règles WSAddressing permet d'adresser des services web et des messages de manière uniforme et indépendamment des systèmes de transport. Utilisez Integration Designer pour sélectionner l'ensemble de règles pour la liaison.

  Remarque : Pour utiliser un ensemble de règles SAML (Security Assertion Markup Language), vous devez effectuer certaines configurations supplémentaires, comme décrit dans «Importation des ensembles de règles SAML», à la page 79.

- SOAP1.1/HTTP

  Utilisez cette liaison si vous souhaitez créer des services Web qui utilisent un message codé par SOAP en fonction de la spécification JAX-RPC (Java API for XML-based RPC).

- SOAP1.1/JMS

  Utilisez cette liaison pour envoyer ou recevoir des messages SOAP à l'aide d'une destination JMS (Java Message Service).

Quel que soit le transport (HTTP ou JMS) utilisé pour transmettre les message SOAP, les liaisons de service Web traitent toujours les interactions de demande/réponse de manière synchrone. L'unité d'exécution à l'origine de l'appel sur le fournisseur de services est bloquée jusqu'à la réception d'une réponse du fournisseur. Pour plus d'informations sur ce style d'appel, voir «Appel synchrone».

Important : Les combinaisons de liaisons de services web suivantes ne peuvent pas être utilisées sur les exportations sur le même module. Si vous devez exposer des composants à l'aide de plusieurs de ces liaisons d'exportation, vous devez placer chacun dans un module distinct, puis connecter ces modules à vos composants à l'aide de la liaison SCA :

- SOAP 1.1/JMS et SOAP 1.1/HTTP à l'aide de JAX-RPC
- SOAP 1.1/HTTP à l'aide de JAX-RPC et SOAP 1.1/HTTP à l'aide de JAX-WS
- SOAP 1.1/HTTP à l'aide de JAX-RPC et SOAP 1.2/HTTP à l'aide de JAX-WS

Une fois que le module SCA contenant la liaison de service Web est déployé sur le serveur, vous pouvez utiliser la console d'administration pour afficher des informations sur la liaison ou pour modifier les propriétés sélectionnées de cette liaison.

Remarque : Les services Web permettent aux applications d'interopérer en utilisant les descriptions standard des services et les formats standard des messages qu'ils échangent. Par exemple, les liaisons d'importation et d'exportation de services Web peuvent interopérer avec des services implémentés à l'aide de Web Services Enhancements (WSE) Version 3.5 et Windows Communication Foundation (WCF) Version 3.5 pour Microsoft .NET. Lors de l'interopération avec de tels services, vous devez veiller à ce que les conditions suivantes soient remplies :

- Le fichier WSDL (Web Services Description Language) utilisé pour accéder à une exportation de service Web inclut une valeur d'action SOAP non vide pour chaque opération exposée dans l'interface.
- Le client de service Web définit l'en-tête SOAPAction ou l'en-tête wsa:Action lorsqu'il envoie des messages à une exportation de service web.

Propagation d'en-têtes SOAP : 

Lors de la gestion des messages SOAP, vous devez accéder aux informations de certains en-têtes SOAP dans les messages reçus. Vérifiez que des messages avec des en-têtes SOAP sont envoyés avec des valeurs spécifiques ou autorisez ces en-têtes SOAP à passer par un module.

Lorsque vous configurez une liaison de services Web dans Integration Designer, vous pouvez indiquer que les en-têtes de SOAP doivent être propagés.

- Lorsque des demandes sont reçues lors d'une exportation ou des réponses sont reçues lors d'une importation, les informations sur les en-têtes SOAP sont accessibles, ce qui permet à la logique dans un module de se baser sur les valeurs d'en-têtes et à ces en-têtes d'être modifiés.
- Lorsque des demandes sont envoyées d'une exportation ou des réponses sont envoyées d'une importation, des en-têtes SOAP peuvent être inclus dans ces messages.

La forme et la présence des en-têtes SOAP propagés peuvent être affectées par les ensembles de règles configurés dans l'importation ou l'exportation, comme décrit dans le tableau 31, à la page 76.

Pour configurer la propagation des en-têtes SOAP pour une importation ou une exportation, vous cliquez (dans la vue Propriétés de Integration Designer) dans l'onglet **Propager l'en-tête de protocole** et sélectionnez les options dont vous avez besoin.

En-tête WS-Addressing

L'en-tête WS-Addressing peut être propagé par la liaison de service Web (JAX-WS).

Quand vous propagez l'en-tête WS-Addressing, prenez en compte les informations suivantes :

- Si vous activez la propagation pour l'en-tête WS-Addressing, celui-ci est propagé dans le module dans les cas suivants :
  - si des demandes sont reçues dans une exportation,
  - si des réponses sont reçues dans une importation.
- L'en-tête WS-Addressing n'est pas propagé dans des messages sortants de IBM Business Process Manager (l'en-tête n'est pas propagé quand des demandes sont envoyées d'une importation ou quand des réponses sont envoyées d'une exportation).

En-tête WS-Security

L'en-tête WS-Security peut être propagé par la liaison de service Web (JAX-WS) et la liaison de service Web (JAX-RPC).

La spécification WS-Security des services Web décrit les améliorations de la messagerie SOAP afin d'offrir une qualité de protection via l'intégrité des messages, leur confidentialité et l'authentification des messages. Ces mécanismes peuvent permettre d'accepter toute une gamme de modèles de sécurité et de technologies de chiffrement.

Quand vous propagez l'en-tête WS-Security, prenez en compte les informations suivantes :
- Si vous activez la propagation pour l'en-tête WS-Security, celui-ci est propagé à travers le module dans les cas suivants :
  - si des demandes sont reçues dans une exportation,
  - si des demandes sont envoyées d'une importation,
  - si des réponses sont reçues dans une importation.
- Par défaut, l'en-tête n'est *pas* propagé quand des réponses sont envoyées d'une exportation Toutefois, si vous définissez la propriété JVM **WSSECURITY.ECHO.ENABLED** à **true**, l'en-tête est propagé quand des réponses sont envoyées de l'exportation. Dans ce cas, si l'en-tête WS-Security dans le chemin de demande n'est pas modifié, les en-têtes WS-Security peuvent être automatiquement répercutés de demandes en réponses.
- La forme exacte du message SOAP envoyé d'une importation pour une demande ou d'une exportation pour une réponse peut ne pas exactement correspondre au message SOAP reçu à l'origine. C'est pourquoi toute signature numérique est censée devenir invalide. Si une signature numérique est obligatoire dans les messages envoyés, elle doit être établie à l'aide d'un ensemble de règles de sécurité approprié, et les en-têtes relatifs à la signature numérique dans les messages reçus doivent être supprimés dans le module.

Pour propager l'en-tête WS-Security, vous devez inclure le schéma WS-Security avec le module d'application. Voir «Inclusion du schéma WS-Security dans un module d'application», à la page 76 pour connaître la procédure pour inclure le schéma.

Mode de propagation des en-têtes

Le mode de propagation des en-têtes dépend de la définition des règles de sécurité sur la liaison d'importation ou d'exportation, comme décrit dans le tableau 31, à la page 76 :

Tableau 31. Mode de transmission des en-têtes de sécurité

	Liaison d'exportation sans règle de sécurité	**Liaison d'exportation avec règle de sécurité**
Liaison d'importation sans règle de sécurité	Les en-têtes de sécurité sont transmis en l'état via le module. Ils ne sont pas déchiffrés. Les en-têtes sont envoyés en sortie sous la même forme qu'ils ont été reçus. La signature numérique peut devenir invalide.	Les en-têtes de sécurité sont déchiffrés et transmis via le module avec la vérification et l'authentification de la signature. Les en-têtes déchiffrés sont envoyés en sortie. La signature numérique peut devenir invalide.
Liaison d'importation avec règle de sécurité	Les en-têtes de sécurité sont transmis en l'état via le module. Ils ne sont pas déchiffrés. Les en-têtes ne doivent pas être propagés vers l'importation. Sinon, une erreur se produit en raison de la duplication.	Les en-têtes de sécurité sont déchiffrés et transmis via le module avec la vérification et l'authentification de la signature. Les en-têtes ne doivent pas être propagés vers l'importation. Sinon, une erreur se produit en raison de la duplication.

Configurez les ensembles de règles appropriés sur les liaisons d'exportation et d'importation, car l'opération épargne le demandeur de service des modifications apportées à la configuration ou aux exigences QoS du fournisseur de services. Si des en-têtes SOAP standard sont visibles dans un module, ils peuvent être utilisés pour influencer le traitement (par exemple, la consignation et le traçage) dans le module. La propagation des en-têtes SOAP à travers le module, d'un message reçu à un message envoyé, signifie que les avantages de l'isolement du module sont moindres.

Les en-têtes standard, comme les en-têtes WS-Security, ne doivent pas être propagés sur une demande vers une importation ou sur une réponse vers une exportation lorsqu'un ensemble de règles, qui entraînerait normalement la génération de ces en-têtes, est associé à une importation ou une exportation. Sinon, une erreur se produit en raison d'une duplication de ces en-têtes. A la place, les en-têtes doivent être explicitement supprimés, ou bien la liaison d'importation ou d'exportation doit être configurée pour empêcher la propagation des en-têtes de protocole.

Accès aux en-têtes SOAP

Quand un message contenant des en-têtes est reçu d'une importation ou d'une exportation de service Web, les en-têtes sont placés dans la section d'en-têtes de l'objet de message de service (SMO). Vous pouvez accéder aux informations d'en-tête, comme décrit dans «Accès aux informations d'en-tête SOAP dans le SMO».

Inclusion du schéma WS-Security dans un module d'application

La procédure suivante montre les étapes pour inclure le schéma dans le module d'application :

- Si l'ordinateur sur lequel Integration Designer s'exécute a accès à Internet, procédez comme suit :
  1. Dans la perspective Intégration métier, sélectionnez **Dépendances** pour votre projet.
  2. Développez **Ressources prédéfinies** et sélectionnez **Fichiers de schéma WS-Security 1.0** ou **Fichiers de schéma WS-Security 1.1** pour importer le schéma dans votre module.
  3. Effacez et régénérez le projet.
- Si l'ordinateur sur lequel Integration Designer s'exécute n'a pas accès à Internet, vous pouvez télécharger le schéma sur un autre ordinateur ayant accès à Internet. Vous pouvez ensuite le copier sur l'ordinateur où Integration Designer s'exécute.

1. Depuis l'ordinateur qui a accès à Internet, téléchargez le schéma distant :
   a. Cliquez sur **Fichier** > **Importer** > **Intégration métier** > **WSDL et XSD**.
   b. Sélectionnez **WSDL distant** ou **Fichier XSD**.
   c. Importez les schémas suivants :

      http://www.w3.org/2003/05/soap-envelope/

      http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd

      http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
2. Copiez les schémas sur l'ordinateur sans accès à Internet.
3. Depuis l'ordinateur sans accès à Internet, importez le schéma :
   a. Cliquez sur **Fichier** > **Importer** > **Intégration métier** > **WSDL et XSD**.
   b. Sélectionnez **WSDL local** ou **Fichier XSD**.
4. Modifiez les emplacements des schémas pour oasis-wss-wssecurity_secext-1.1.xsd :
   a. Ouvrez le schéma dans *emplacement_espace_de_travail*/*nom_module*/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd.
   b. Remplacez :

      ```
      <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
      schemaLocation='http://www.w3.org/2003/05/soap-envelope/'/>
      ```

      par :

      ```
      <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
      schemaLocation='../w3/_2003/_05/soap_envelope.xsd'/>
      ```
   c. Remplacez :

      ```
      <xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
      schemaLocation='http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd'/>
      ```

      par :

      ```
      <xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
      schemaLocation='../w3/tr/_2002/rec_xmlenc_core_20021210/xenc-schema.xsd'/>
      ```
5. Modifiez l'emplacement du schéma pour oasis-200401-wss-wssecurity-secext-1.0.xsd :
   a. Ouvrez le schéma dans *emplacement_espace_de_travail*/*nom_module*/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd.
   b. Remplacez :

      ```
      <xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
      schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
      ```

      par :

      ```
      <xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
      schemaLocation="../w3/tr/_2002/rec_xmldsig_core_20020212/xmldsig-core-schema.xsd"/>
      ```
6. Effacez et régénérez le projet.

Propagation d'en-têtes de transport : Advanced

Lors de la gestion des messages SOAP, vous devez accéder aux informations de certains en-têtes de transport dans les messages reçus. Vérifiez que des messages avec des en-têtes de transport sont envoyés avec des valeurs spécifiques ou autorisez ces en-têtes de transport à passer par un module.

Lorsque vous configurez une liaison de services Web dans Integration Designer, vous pouvez indiquer que les en-têtes de transport doivent être propagés.

- Lorsque des demandes sont reçues lors d'une exportation ou des réponses sont reçues lors d'une importation, les informations sur les en-têtes de transport sont accessibles, ce qui permet à la logique dans un module de se baser sur les valeurs d'en-têtes et à ces en-têtes d'être modifiés.
- Lorsque des réponses sont envoyées d'une exportation ou des demandes sont envoyées d'une importation, des en-têtes de transport peuvent être inclus dans ces messages.

Spécification d'une propagation d'en-têtes

Pour configurer la propagation des en-têtes de transport pour une importation ou une exportation, procédez comme suit :

1. Dans la vue Propriétés de Integration Designer, sélectionnez **Liaison** > **Propagation**.
2. Définissez l'option de propagation d'en-têtes de votre choix.

Remarque : Par défaut, la propagation d'en-têtes de transport est désactivée et ne peut être déployée que dans un environnement d'exécution Version 7.0.0.3 (ou ultérieure). Notez aussi que pour la Version 7.0.0.3, la propagation d'en-têtes de transport est limitée à des en-têtes de transport HTTP uniquement.

Si vous activez une propagation des en-têtes de transport, les en-têtes seront propagés sur un module à partir de messages reçus et, si vous ne supprimez pas explicitement les en-têtes, ceux-ci seront utilisés dans les appels suivants dans la même unité d'exécution.

Remarque : Les en-têtes de transport ne peuvent pas être propagés si vous utilisez la liaison de services web (JAX-RPC).

Accès aux informations d'en-tête

Si la propagation d'en-têtes de transport est activée pour des messages reçus, tous les en-têtes de transport (y compris des en-têtes définis par le client) sont visibles dans l'objet de message de service (SMO). Vous pouvez définir les en-têtes pour différentes valeurs ou en créer d'autres. Notez toutefois qu'il n'existe aucun contrôle ni validation des valeurs que vous définissez et que tout en-tête incorrect peut causer des problèmes lors de l'exécution du service Web.

Prenez en compte les informations suivantes lorsque vous définissez des en-têtes HTTP :

- Tout changement apporté aux en-têtes réservés pour un moteur de service Web ne sera pas repris dans le message sortant. Par exemple, la version ou méthode HTTP, le contenu type, la longueur du contenu et les en-têtes SOAPAction sont réservés au moteur de service Web.
- Si la valeur d'en-tête est un nombre, le nombre (contrairement à la chaîne) doit être défini directement. Par exemple, utilisez **Max-Forwards = 5** (au lieu de **Max-Forwards = Max-Forwards: 5**) et **Age = 300** (au lieu de **Age = Age: 300**).
- Si la taille du message de demande ne dépasse pas 32 Ko, le moteur de services Web supprime l'en-tête Transfert-Codage et définit à la place l'en-tête Longueur-contenu avec la taille fixe du message.
- Le Langage-Contenu est réinitialisé par WAS.channel.http sur le chemin de réponse.
- Un paramètre de mise à niveau non valide se traduit par une erreur 500.
- Les en-têtes suivants ajoutent la valeur réservée par le moteur de service Web aux paramètres du client :
  - Agent d'utilisateur
  - Contrôle de cache
  - Pragma
  - Accepter
  - Connexion

Vous pouvez accéder aux informations d'en-tête de l'une des manières suivantes :

- En utilisant une primitive de médiation pour accéder à la structure SMO.

  Voir les liens «Rubriques connexes» pour retrouver des informations sur l'utilisation des primitives de médiation.
- En utilisant une interface SPI des services de contexte

  Le code exemple suivant lit des en-têtes de transport HTTP à partir de services de contexte :

```
HeadersType headerType = ContextService.INSTANCE.getHeaders();
HTTPHeaderType httpHeaderType = headerType.getHTTPHeader();
List HTTPHeader  httpHeaders = httpHeaderType.getHeader();
if(httpHeaders!=null){
 for(HTTPHeader httpHeader: httpHeaders){
   String httpHeadername = httpHeader.getName();
   String httpHeaderValue = httpHeader.getValue();
     }
}
List PropertyType  properties = headerType.getProperties();
if(properties!=null){
  for(PropertyType property: properties){
   String propertyName = property.getName();
   String propertyValue = property.getValue().toString();
     }
}
```

Identification et résolution des incidents

En cas de problèmes lors de l'envoi des en-têtes révisés, vous pouvez intercepter le message TCP/IP en utilisant des outils comme Moniteur TCP/IP dans Integration Designer. Pour accéder au Moniteur TCP/IP, dans la page Préférence, sélectionnez **Run/Debug** > **Moniteur TCP/IP**.

Vous pouvez aussi visualiser les valeurs d'en-tête à l'aide de la trace du moteur JAX-WS : **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Utilisation des liaisons de services Web (JAX-WS) : Advanced

Lorsque vous utilisez des liaisons de services Web (JAX-WS) avec vos applications, vous pouvez ajouter une qualité de service (QOS) SAML (Security Assertion Markup Language) à la liaison. Vous devez tout d'abord utiliser la console d'administration pour importer l'ensemble de règles. Vous pouvez également utiliser la console d'administration pour vous assurer que le serveur est correctement configuré pour être utilisé avec la liaison de services Web (JAX-WS).

Importation des ensembles de règles SAML : Advanced

Norme OASIS basée sur XML, SAML (Security Assertion Markup Language) permet l'échange d'informations entre des attributs d'identité de l'utilisateur et de sécurité. Lorsque vous configurez une liaison de services Web (JAX-WS) dans Integration Designer, vous pouvez indiquer un ensemble de règles SAML. A l'aide de la console d'administration d'IBM Business Process Manager, commencez par rendre les ensembles de règles disponibles pour permettre leur importation dans Integration Designer.

Les ensembles de règles SAML se trouvent généralement dans le répertoire de configuration des profils :

racine_profil/config/templates/PolicySets

Avant de lancer cette procédure, vérifiez que les répertoires suivants (qui contiennent les ensembles de règles) se trouvent bien dans le répertoire de configuration des profils :

- SAML11 Bearer WSHTTPS default
- SAML20 Bearer WSHTTPS default
- SAML11 Bearer WSSecurity default
- SAML20 Bearer WSSecurity default
- SAML11 HoK Public WSSecurity default
- SAML20 HoK Public WSSecurity default
- SAML11 HoK Symmetric WSSecurity default
- SAML20 HoK Symmetric WSSecurity default

- `Username WSHTTPS default`

Si les répertoires ne figurent pas dans le répertoire de configuration des profils, copiez-les dans ce répertoire à partir de l'emplacement suivant :

racine_serveur_app`/profileTemplates/default/documents/config/templates/PolicySets`

Importez les ensembles de règles dans la console d'administration, sélectionnez ceux que vous voulez rendre accessibles à Integration Designer, pour sauvegardez un fichier .zip pour chacun de ces ensembles de règles dans un emplacement accessible par Integration Designer.

1. Importez les ensembles de règles en procédant comme suit :
   a. Dans la console d'administration, cliquez sur **Services** > **Ensemble de règles** > **Ensembles de règles de l'application**.
   b. Cliquez sur **Importer** > **A partir du référentiel par défaut**.
   c. Sélectionnez l'ensemble de règles par défaut SAML, puis cliquez sur **OK**.
2. Exportez les ensembles de règles pour qu'ils puissent être utilisés par Integration Designer:
   a. Sur la page Ensembles de règles de l'application, sélectionnez l'ensemble de règles SAML à exporter, puis cliquez sur **Exporter**.

      Remarque : Si la page Ensembles de règles de l'application n'est pas actuellement affichées, sur la console d'administration, cliquez sur **Services** > **Ensembles de règles** > **Ensembles de règles de l'application**.
   b. Sur la page suivante, cliquez sur le lien du fichier .zip pour l'ensemble de règles.
   c. Dans la fenêtre de téléchargement de fichier, cliquez sur **Sauvegarder** et indiquez un emplacement accessible par Integration Designer.
   d. Cliquez sur **Retour**.
   e. Complétez les étapes 2a via 2d pour chaque ensemble de règles à exporter.

Les ensembles de règles sont enregistrés dans des fichiers .zip et sont prêts à être exportés dans Integration Designer.

Importez les ensembles de règles dans Integration Designer, comme décrit dans la rubrique «Ensembles de règles».

Appel des services Web utilisant l'authentification de base HTTP : Advanced

Le procédure d'authentification de base HTTP demande de saisir un nom d'utilisateur et un mot de passe pour authentifier un client de service sur un noeud final sécurisé. Vous pouvez configurer l'authentification de base HTTP lorsque vous envoyez ou que vous recevez des requêtes de service Web.

Pour configurer l'authentification de base HTTP afin de recevoir des requêtes de service Web, vous devez configurer la liaison d'exportation de l'API Java des services Web XML (JAX-WS) comme indiqué dans Création et affectation de rôles de sécurité aux exportations de service Web.

Vous pouvez activer l'authentification de base HTTP pour les requêtes de service Web envoyées par une liaison d'importation de l'API Java des services Web XML de deux manières :

- Lorsque vous configurez la liaison d'importation dans un module SCA, vous pouvez sélectionner l'ensemble de règles HTTP nommé BPMHTTPBasicAuthentication (fourni avec la liaison d'importation de service Web JAX-WS) ou un autre ensemble de règles contenant la règle HTTPTransport.
- Lorsque vous construisez le module SCA, vous pouvez utiliser des fonctions de flux de médiation pour créer dynamiquement un en-tête d'authentification HTTP et spécifier les données du nom d'utilisateur et du mot de passe dans cet en-tête.

Remarque : L'ensemble de règles est prioritaire sur la valeur indiquée dans l'en-tête. Si vous souhaitez utiliser la valeur définie dans l'en-tête d'authentification HTTP pendant la phase d'exécution, n'associez pas un ensemble de règles comprenant la règle HTTPTransport. En particulier, n'utilisez pas l'ensemble de règles par défaut BPMHTTPBasicAuthentication et, si vous avez défini un autre ensemble de règles, vérifiez qu'il ne comprend pas la règle HTTPTransport.

Pour plus d'informations sur les liaisons de règles, les ensembles de règles de service web et leur utilisation, voir Ensembles de règles de service Web du centre de documentation de WebSphere Application Server.

- Pour utiliser l'ensemble de règles fourni, exécutez les opérations suivantes :
  1. Facultatif : Dans la console d'administration, créez une liaison de règles générales de client ou éditez une liaison de règles existante qui contienne la règle HTTPTransport avec les valeurs requises pour l'ID utilisateur et le mot de passe.
  2. Dans IBM Integration Designer, générez une liaison d'importation de service Web (JAX-WS) et associez-y l'ensemble de règles BPMHTTPBasicAuthentication.
  3. Exécutez l'*une* des actions suivantes :
     - Dans IBM Integration Designer, dans les propriétés de la liaison d'importation de service Web (JAX-WS), indiquez le nom d'une liaison de règles générales de client existante comprenant la règle HTTPTransport.
     - Après avoir déployé le module SCA, utilisez la console d'administration pour sélectionner une liaison de règles de client existante ou créer une nouvelle liaison de règles de client et l'associer à la liaison d'importation.
  4. Facultatif : Dans la console d'administration du Process Server, éditez la liaison d'ensemble de règles pour y spécifier l'ID et le mot de passe requis.
- Pour indiquer le nom d'utilisateur et le mot de passe dans l'en-tête d'authentification HTTP, exécutez l'une des procédures suivantes :
  - Utilisez la primitive de médiation HTTP Header Setter dans IBM Integration Designer afin de créer l'en-tête d'authentification HTTP, puis indiquez le nom d'utilisateur et le mot de passe.
  - S'il faut ajouter une logique supplémentaire, insérez un code Java dans une primitive de médiation personnalisée (comme dans l'exemple suivant) pour :
    1. Créer l'en-tête d'authentification HTTP
    2. Indiquer les données du nom d'utilisateur et du mot de passe
    3. Ajouter le nouvel en-tête d'authentification HTTP à la règle HTTPControl
    4. Redéfinir la règle HTTPControl après sa mise à jour dans le service de contexte

```
//Get the HeaderInfoType from contextService
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Get the HTTP header and HTTP Control from HeaderInfoType
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Create new HTTPAuthentication and set the HTTPCredentials
HTTPAuthentication authorization = factory.createHTTPAuthentication();
HTTPCredentials credentials = factory.createHTTPCredentials();
authorization.setAuthenticationType(HTTPAuthenticationType.BASIC_LITERAL);
credentials.setUserId("USERNAME");
credentials.setPassword("PASSWORD");
authorization.setCredentials(credentials);
```

```
cp.setAuthentication(authorization);
httpHeaderType.setControl(cp);
// Set header info back to the current execution context.
contextService.setHeaderInfo(headers);
```

Vérification de la configuration du serveur : Advanced

Lorsque vous déployez une application avec une liaison de service Web (JAX-WS), vérifiez que l'option **Démarrer les composants en fonction des besoins** du serveur cible sur laquelle l'application est déployée n'est pas sélectionnée.

Pour vérifier cela, procédez comme suit à partir de la console d'administration :

1. Cliquez sur **Serveurs** > **Types de serveurs** > **Serveurs d'applications WebSphere**.
2. Cliquez sur le nom du serveur.
3. Sous l'onglet Configuration, vérifiez si l'option **Démarrer les composants en fonction des besoins** est sélectionnée.
4. Exécutez l'une des étapes suivantes :
   - Si l'option **Démarrer les composants en fonction des besoins** est sélectionnée, désactivez la case à cocher, puis cliquez sur **Appliquer**.
   - Si l'option **Démarrer les composants en fonction des besoins** n'est pas sélectionnée, cliquez sur **Annuler**.

Pièces jointes dans les messages SOAP : Advanced

Vous pouvez envoyer et recevoir des messages SOAP qui incluent des données binaires (fichiers PDF ou images JPEG) comme pièces jointes. Les pièces jointes peuvent être *référencées* (à savoir, représentées explicitement comme parties de message dans l'interface de service) ou *non référencées* (auquel cas, un nombre et des types arbitraires de pièces jointes peuvent être incluses).

Une pièce jointe référencée peut être représentée de l'une des manières suivantes :

- Les pièces jointes MTOM utilisent le codage SOAP spécifié par Message Transmission Optimization Mechanism (http://www.w3.org/TR/soap12-mtom/). Les pièces jointes MTOM sont activées par l'intermédiaire d'une option de configuration dans les liaisons d'importation et d'exportation, et doivent être utilisées pour coder les pièces jointes des nouvelles applications.
- Comme élément wsi:swaRef-typed dans le schéma du message

  Les pièces jointes définies à l'aide du type wsi:swaRef sont conformes à Web Services Interoperability Organization (WS-I) *Attachments Profile Version 1.0* (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), qui définit la manière dont les éléments de message sont associés aux composants MIME.
- Comme composant de message de niveau supérieur, à l'aide d'un schéma de type binaire

  Les pièces jointes représentées sous forme de composants de message de niveau supérieur respectent la spécification des *messages SOAP avec pièces jointes* (http://www.w3.org/TR/SOAP-attachments).

  Vous pouvez aussi configurer des pièces jointes représentées comme des parties de message de niveau supérieur pour vous assurer que le document WSDL et les messages produits par la liaison sont conformes à WS-I *Attachments Profile Version 1.0* et à WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

Une pièce jointe non référencée est transportée dans un message SOAP sans aucune représentation dans le schéma du message.

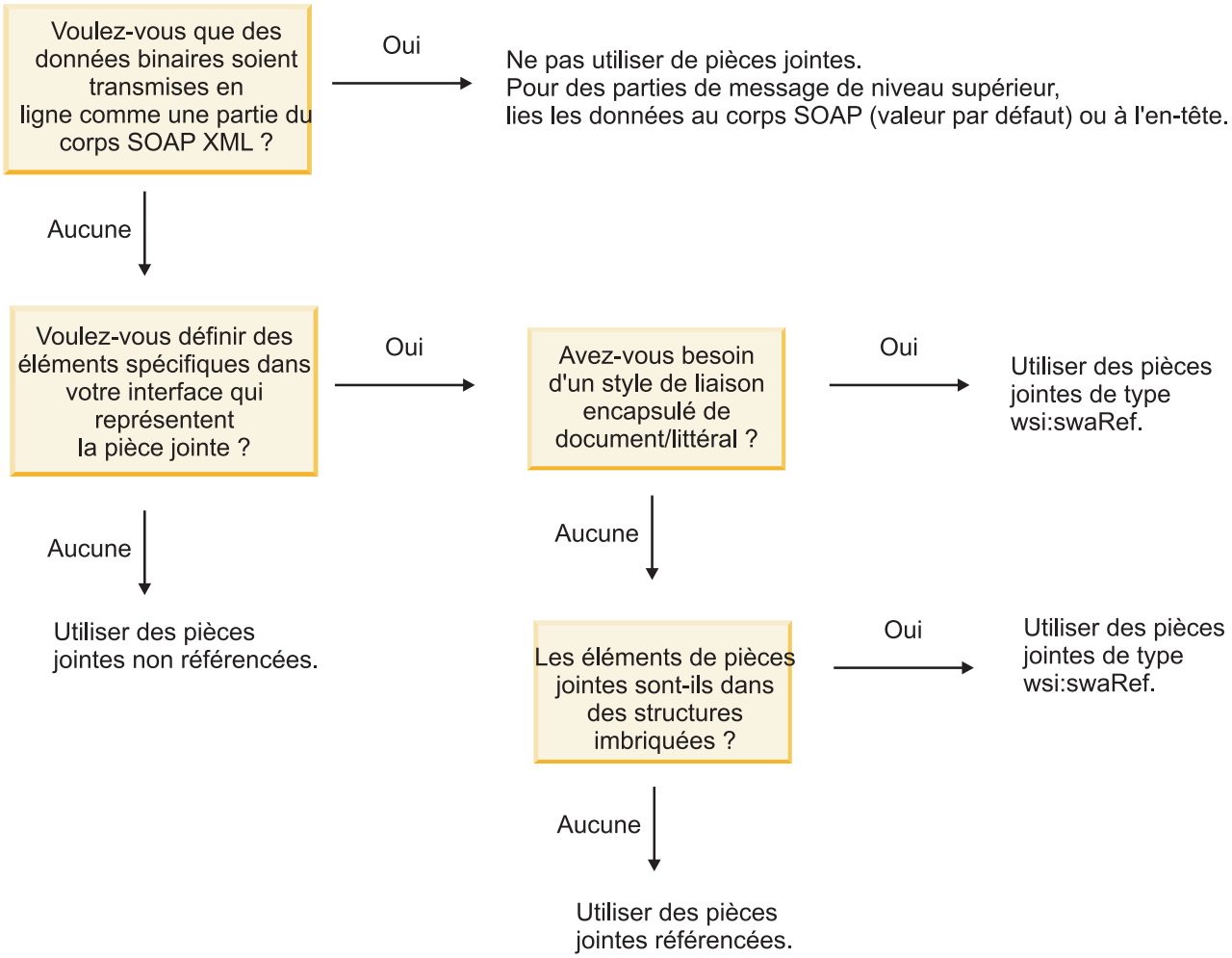
Dans tous les cas, sauf pour les pièces jointes MTOM, la liaison SOAP WSDL doit inclure une liaison MIME pour que des pièces jointes puissent être utilisées et la taille maximale de ces pièces jointes ne doit pas dépasser 20 Mo.

Remarque : Pour envoyer ou recevoir des messages SOAP avec des pièces jointes, vous devez utiliser l'une des liaisons de service Web basée sur JAX-WS (Java API for XML Web Services).

Choix du style de pièces jointe appropriées : Advanced

Lors de la conception d'une interface de service qui inclut des données binaires, envisagez la façon dont les données binaires sont traitées dans les messages SOAP qui sont envoyés et reçus par le service.

Le mécanisme MTOM (Message Transmission Optimization Mechanism) doit être utilisé pour les pièces jointes si l'application de service Web connectée le prend en charge. Si tel n'est pas le cas, le diagramme suivant indique comment les autres styles de pièces jointes sont choisis. A l'aide du jeu de questions suivant déterminez le style de pièces jointes approprié :



Pièces jointes MTOM : composants de message de niveau supérieur :

Vous pouvez envoyer et recevoir des messages de service Web qui incluent des pièces jointes SOAP MTOM (Message Transmission Optimization Mechanism). Dans un message SOAP à plusieurs parties MIME, le corps SOAP constitue la première partie du message et la ou les pièces jointes sont dans les parties suivantes.

Lorsque vous envoyez ou recevez une pièce jointe référencée dans un message SOAP, les données binaires qui constituent la pièce jointe (dont la taille est souvent importante) sont conservées séparément

du corps du message SOAP pour qu'elles n'aient pas besoin d'être analysées comme des données XML. Le traitement est donc plus efficace que si les données binaires étaient conservées dans un élément XML.

Voici un exemple de message SOAP MTOM :

```
... autres en-têtes de transport ...
Content-Type: multipart/related; boundary=MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812;
type="application/xop+xml"; start="
<0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>"; start-info="text/xml"; charset=UTF-8

--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: application/xop+xml; charset=UTF-8; type="text/xml";
content-transfer-encoding: binary
content-id:
   <0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header/>
      <soapenv:Body>
         <sendImage xmlns="http://org/apache/axis2/jaxws/sample/mtom">
            <input>
            <imageData><xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
            href="cid:1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org"/></imageData>
            </input>
         </sendImage>
      </soapenv:Body>
   </soapenv:Envelope>
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: text/plain
content-transfer-encoding: binary
content-id:
         <1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org>

... emplacement pour les données binaires ...
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812--
```

Notez que, dans cet exemple de message MTOM, l'élément content-type de l'enveloppe SOAP est **application/xop+xml** et que les données binaires sont remplacées par un élément **xop:Include** similaire au suivant :

```
<xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
href="cid:1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org"/>
```

Traitement entrant des pièces jointes référencées

Lorsqu'un client transmet un message SOAP avec une pièce jointe à un composant SCA (Service Component Architecture), la liaison d'exportation de service Web (JAX-WS) commence par supprimer cette pièce jointe. Elle analyse ensuite la partie SOAP du message et crée un objet métier. Enfin, la liaison définit le code binaire de la pièce jointe dans l'objet métier.

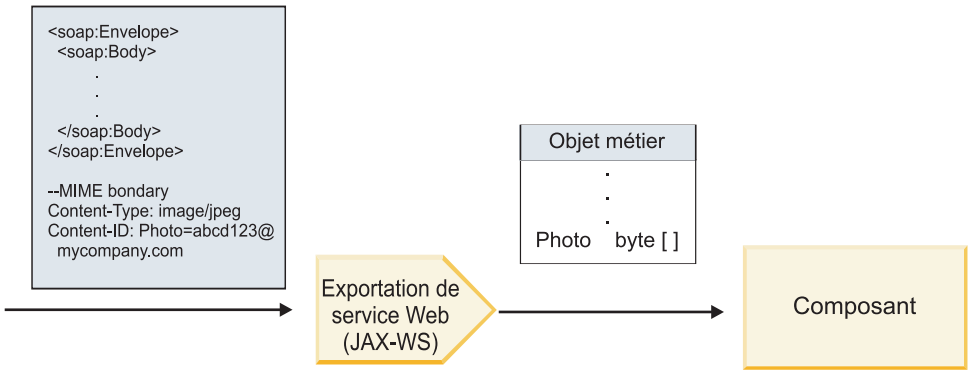


Figure 13. Mode de traitement d'un message SOAP avec une pièce jointe référencée par la liaison d'exportation de service Web (JAX-WS)

Attributs de pièce jointe MTOM

- MTOM peut prendre en charge les pièces jointes dans des structures imbriquées.
- MTOM est uniquement disponible pour le type base64Binary.
- MTOM peut prendre en charge les pièces jointes dans les structures imbriquées, cela signifiant que l'élément **bodyPath** associé aux pièces jointes MTOM est l'emplacement **xpath** de l'élément dans lequel la pièce jointe MTOM est contenue. La logique de calcul de **bodyPath** suit strictement le schéma permettant de générer l'emplacement **xpath** comme indiqué dans les exemples ci-après :
  - Pour un type autre que le tableau (**maxOccurs** a pour valeur 1) : /sendImage/input/imageData
  - Pour un type tableau (**maxOccurs** > 1) : /sendImage/input/imageData[1]
- Les types de pièces jointes mixtes ne sont pas pris en charge, cela signifiant que si MTOM est activé pour la liaison d'importation, la pièce jointe MTOM sera générée. Si MTOM est désactivé ou si la valeur de la configuration MTOM est la valeur par défaut pour la liaison d'exportation, le message MTOM entrant n'est pas pris en charge.

Pièces jointes référencées : éléments de type swaRef : Advanced

Vous pouvez envoyer et recevoir des messages SOAP qui incluent des pièces jointes représentées dans l'interface des services comme des éléments de type swaRef.

Un élément de type swaRef est défini dans Web Services Interoperability Organization (WS-I) *Attachments Profile* Version 1.0 (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), qui définit la manière dont les éléments de message sont associés aux composants MIME.

Dans le message SOAP, le corps du message SOAP contient un élément de type swaRef qui identifie l'ID contenu de la pièce jointe.

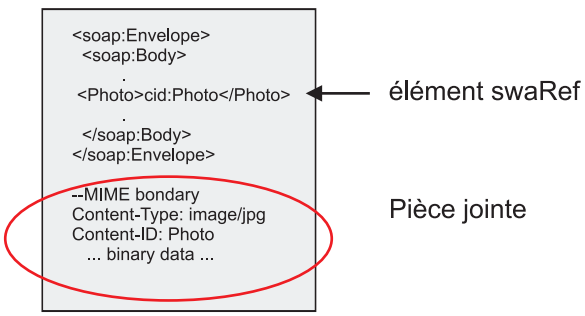


Figure 14. Message SOAP avec élément swaRef

Le WSDL de ce message SOAP contient un élément de type swaRef dans une portion de message qui identifie la pièce jointe.

```
<element name="sendPhoto">
  <complexType>
     <sequence>
       <element name="Photo" type="wsi:swaRef"/>
     </sequence>
  </complexType>
 </element>
```

Le WSDL doit également contenir une liaison MIME qui indique que les messages MIME composite doivent être utilisés.

Remarque : Le WSDL n'inclut *pas* de liaison MIME pour l'élément de message de type swaRef spécifique, car les liaisons MIME ne s'appliquent qu'aux portions de message de niveau supérieur.

La propagation des pièces jointes représentées comme des éléments de type swaRef n'est possible qu'entre composants de flux de médiation. Si un autre type de composant doit accéder à la pièce jointe ou servir de cible lors de la propagation de celle-ci, utilisez un composant de flux de médiation pour déplacer la pièce jointe vers un emplacement accessible par le composant.

Traitement entrant des pièces jointes

Utilisez Integration Designer pour configurer une liaison d'exportation afin de recevoir la pièce jointe. Vous créez un module, ainsi que son interface et ses opérations associées, et notamment un élément de type swaRef. Créez ensuite une liaison de service Web (JAX-WS).

Remarque : Pour obtenir des informations détaillées, reportez-vous à la rubrique «Utilisation des pièces jointes» du centre de documentation de Integration Designer.

Lorsqu'un client transmet un message SOAP avec une pièce jointe swaRef à un composant SCA (Service Component Architecture), la liaison d'exportation de service Web (JAX-WS) commence par supprimer cette pièce jointe. Elle analyse ensuite la partie SOAP du message et crée un objet métier. Enfin, la liaison définit l'ID contenu de la pièce jointe dans l'objet métier.

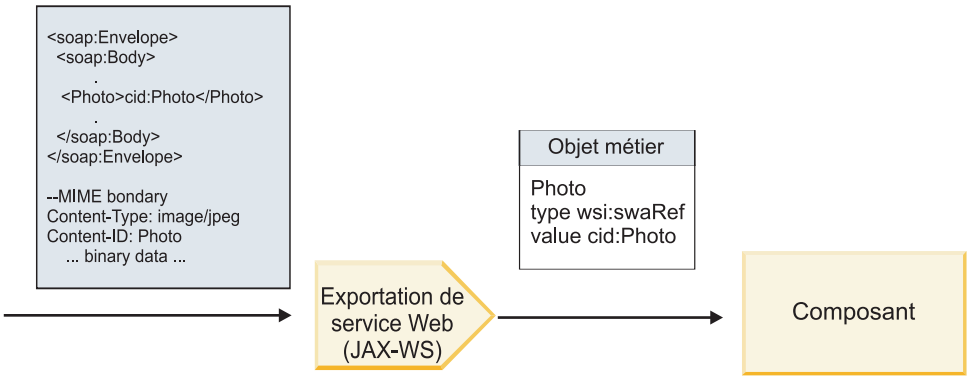


Figure 15. Méthode de traitement d'un message SOAP avec une pièce jointe swaRef par la liaison d'exportation de service Web (JAX-WS)

Accès aux métadonnées de pièce jointe dans un composant de flux de médiation

Comme illustré dans la figure 16, à la page 87, lorsque des composants accèdent à des pièces jointes swaRef, l'identificateur de contenu de la pièce jointe apparaît comme élément de type swaRef.

Chaque pièce jointe d'un message SOAP contient également un élément **attachments** correspondant dans l'objet SMO. Lorsqu'il utilise le type WS-I swaRef, l'élément **attachments** inclut le type de contenu de la pièce jointe et l'ID contenu, ainsi que les données binaires de la pièce jointe.

Pour obtenir la valeur d'une pièce jointe swaRef, il est donc nécessaire d'obtenir la valeur de l'élément de type swaRef, puis de rechercher l'élément **attachments** avec la valeur **contentID** correspondante. Notez que pour la valeur **contentID**, le préfixe **cid:** est généralement supprimé de la valeur swaRef.

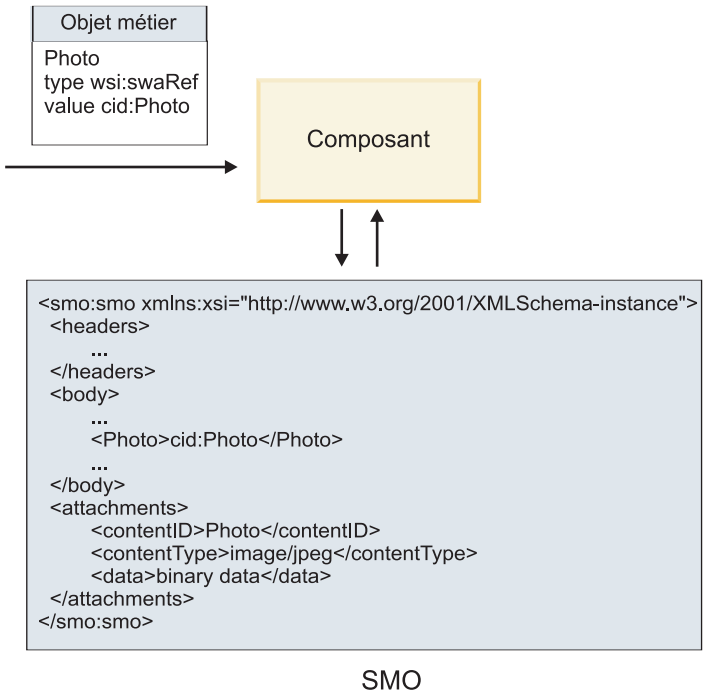


Figure 16. Affichage des pièces jointes swaRef dans l'objet SMO

Traitement sortant

Utilisez Integration Designer pour configurer une liaison d'importation de service Web (JAX-WS) de sorte qu'elle appelle un service Web externe. La liaison d'importation est configurée avec un document WSDL qui décrit le service Web à appeler et définit la pièce jointe qui sera transmise au service Web.

Lorsqu'un message SCA est reçu par une liaison d'importation de service Web (JAX-WS), les éléments de type swaRef sont envoyés comme pièces jointes si l'importation est connectée à un composant de flux de médiation et que l'élément de type swaRef possède un élément **attachments** correspondant.

Pour le traitement sortant, les éléments de type swaRef sont toujours envoyés avec leur ID contenu, mais le module de médiation doit s'assurer qu'il existe un élément **attachments** correspondant avec une valeur **contentID** correspondante.

Remarque : Pour se conformer au profil des pièces jointes WS-I, la valeur **content ID** doit suivre le "codage des portions content-id," comme décrit dans la section 3.8 de WS-I *Attachments Profile* 1.0.

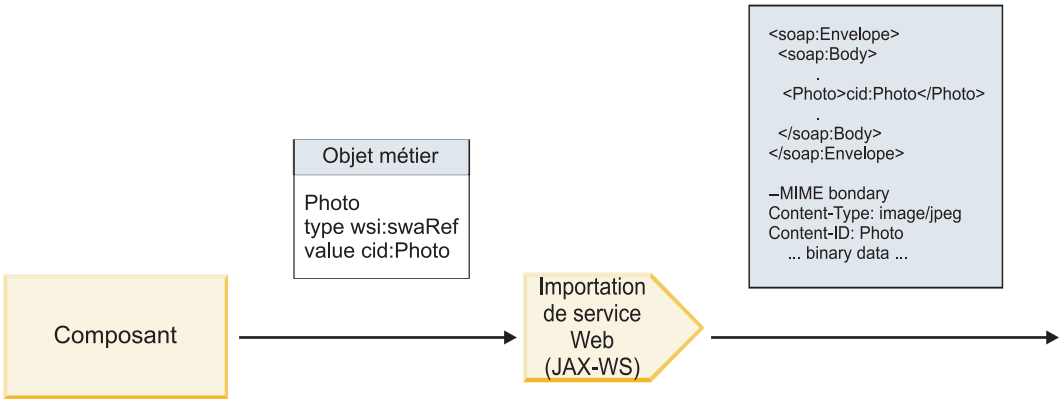


Figure 17. Méthode de génération d'un message SOAP avec une pièce jointe swaRef par la liaison d'importation de service Web (JAX-WS)

Définition des métadonnées de pièce jointe dans un composant de flux de médiation

Si, dans l'objet SMO, il existe une valeur d'élément de type swaRef et un élément **attachments**, la liaison prépare le message SOAP (avec la pièce jointe) et l'envoie à un destinataire.

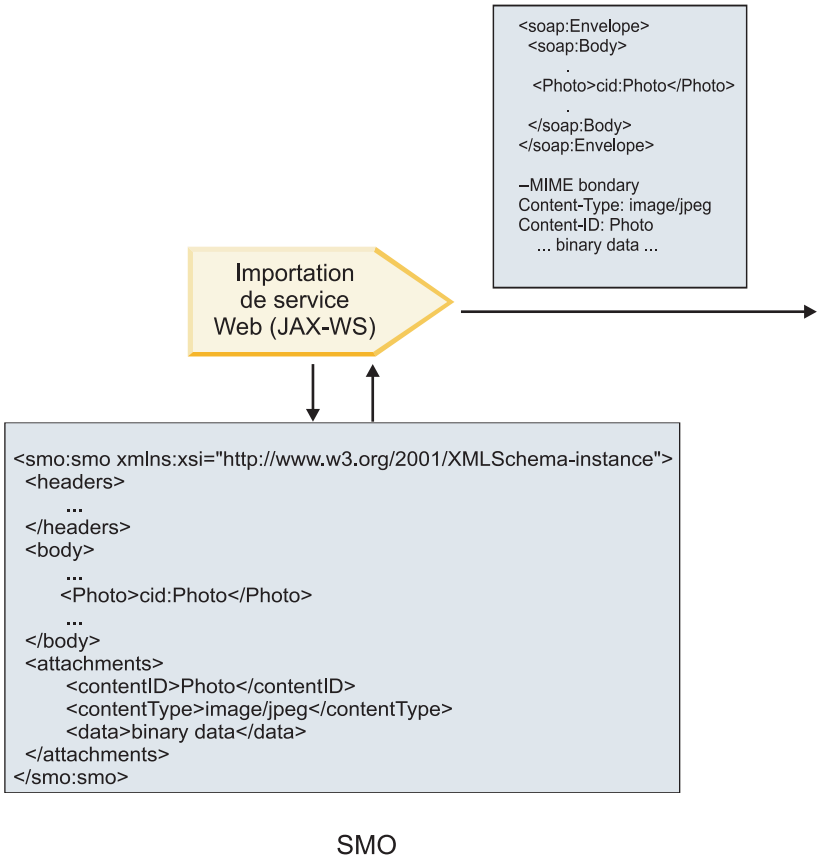


Figure 18. Méthode d'accès à une pièce jointe swaRef de l'objet SMO pour créer le message SOAP

L'élément **attachments** n'est présent dans l'objet SMO que si un composant de flux de médiation est connecté directement à l'importation ou l'exportation ; il n'est pas transmis entre les autres types de composant. Si les valeurs sont requises dans un module contenant d'autres types de composants, un composant de flux de médiation doit être utilisé pour copier ces valeurs dans un emplacement où elles

sont accessibles dans le module et un autre composant de flux de médiation doit être utilisé pour définir les valeurs correctes avant un appel sortant par l'intermédiaire d'une importation de service Web.

Important : Comme décrit dans la «représentation XML de l'objet SMO,» la primitive de médiation de mappage convertit les messages à l'aide d'une transformation XSLT 1.0. La transformation agit sur une sérialisation XML de l'objet SMO. La primitive de médiation de Mappage permet de spécifier la racine de la sérialisation et l'élément racine du document XML reflète cette racine.

Lorsque vous envoyez des messages SOAP comportant des pièces jointes, l'élément racine choisi détermine le mode de propagation des pièces jointes.

- Si vous utilisez «/body» comme racine de la mappe XML, toutes les pièces jointes sont propagées dans la mappe par défaut.
- Si vous utilisez «/» comme racine de la mappe, vous pouvez contrôler la propagation des pièces jointes.

Pièces jointes référencées : composants de message de niveau supérieur : Advanced

Vous pouvez envoyer et recevoir des messages SOAP qui incluent des pièces jointes binaires déclarées comme composants dans votre interface de service.

Dans un message SOAP à plusieurs parties MIME, le corps SOAP constitue la première partie du message et les pièces jointes sont dans les parties suivantes.

Quel est l'avantage d'envoyer ou de recevoir une pièce jointe référencée dans un message SOAP ? Les données binaires qui constituent la pièce jointe (dont la taille est souvent importante) sont conservées séparément du corps du message SOAP pour qu'elles n'aient pas besoin d'être analysées comme des données XML. Le traitement est donc plus efficace que si les données binaires étaient conservées dans un élément XML.

Types de messages SOAP avec pièces jointes référencées

A partir de la Version 7.0.0.3 de IBM Business Process Manager, vous avez le choix du mode de génération du message SOAP :

- **Messages compatibles WS-I**

  L'exécution peut générer des messages SOAP conformes à WS-I *Attachments Profile Version 1.0* et à WS-I *Basic Profile Version 1.1*. Dans un message SOAP conforme à ces profils, une seule partie du message est liée au corps SOAP ; pour celles liées en tant que pièces jointes, on utilise le codage des portions content-id (comme décrit dans WS-I *Attachments Profile Version 1.0*) pour lier la pièce jointe à la partie message.
- **Messages non compatibles WS-I**

  L'exécution peut générer des messages SOAP non conformes aux profils WS-I, mais qui sont compatibles avec les messages générés dans la Version 7.0 ou 7.0.0.2 de IBM Business Process Manager. Les messages SOAP utilisent des éléments de niveau supérieur nommés après la partie message avec un attribut **href** qui retient la pièce jointe **content-id**, mais le codage des portions content-id (comme décrit dans WS-I *Attachments Profile Version 1.0*) n'est pas utilisé.

Sélection de la compatibilité WS-I pour les exportations de services Web

Utilisez Integration Designer pour configurer une liaison d'exportation. Créez un module, ainsi que son interface et ses opérations associées. Créez ensuite une liaison de service Web (JAX-WS). La page Pièces jointes référencées affiche toutes les portions binaires de l'opération créée et vous sélectionnez les parties qui seront des pièces jointes. Indiquez ensuite, sur la page Specify the WS-I AP 1.0 compliance (Spécifier la compatibilité WS-I AP 1.0) de Integration Designer, l'un des choix suivants :

- **Use WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP compatible WS-I AP 1.0)**

Si vous sélectionnez cette option, vous indiquez aussi quelle partie du message doit être liée au corps SOAP.

Remarque : Cette option ne peut être utilisée que si le fichier WSDL correspondant est également compatible WS-I.

Un fichier WSDL généré par Integration Designer Version 7.0.0.3 est compatibles avec WS-I. Toutefois, vous ne pouvez pas sélectionner cette option, si vous importez un fichier WSDL qui n'est pas compatible avec WS-I.

- **Use non WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP non compatible WS-I AP 1.0)**

  Si vous sélectionnez cette option, qui est celle par défaut, la première partie du message est liée au corps SOAP.

Remarque : Seules les portions de message de niveau supérieur (les éléments définis dans le type de port WSDL (WSDL portType) comme portions du message en entrée ou en sortie) dont le type est binaire (base64Binary ou hexBinary) peuvent être envoyées ou reçues comme pièces jointes référencées. Pour obtenir des informations détaillées, reportez-vous à la rubrique «Utilisation des pièces jointes» du centre de documentation de Integration Designer.

Pour des messages compatibles WS-I, le content-ID qui est généré dans le message SOAP est la concaténation des éléments suivants :

- La valeur de l'attribut **nom** de l'élément **wsdl:part** référencé par **mime:content**
- Le caractère **=**
- Une valeur globalement unique, comme un UUID (identificateur unique universel)
- Le caractère **@**
- Un nom de domaine valide

Traitement entrant des pièces jointes référencées

Lorsqu'un client transmet un message SOAP avec une pièce jointe à un composant SCA (Service Component Architecture), la liaison d'exportation de service Web (JAX-WS) commence par supprimer cette pièce jointe. Elle analyse ensuite la partie SOAP du message et crée un objet métier. Enfin, la liaison définit le code binaire de la pièce jointe dans l'objet métier.

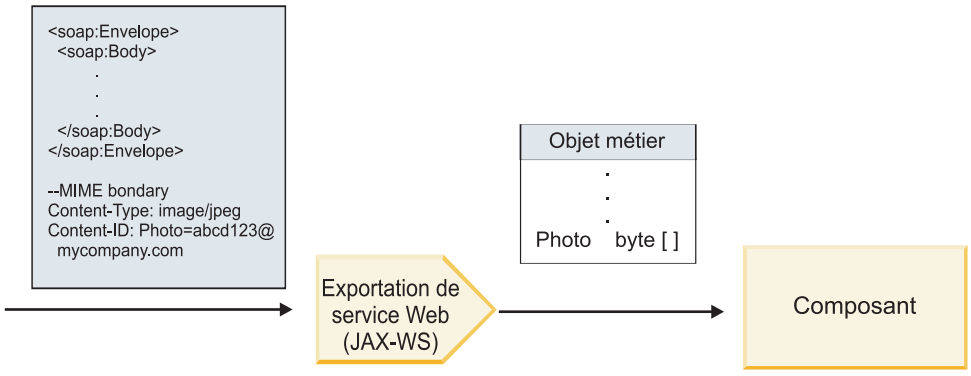


Figure 19. Mode de traitement d'un message SOAP compatible WS-I avec une pièce jointe référencée par la liaison d'exportation de service Web (JAX-WS)

Accès aux métadonnées de pièce jointe dans un composant de flux de médiation

Comme illustré dans la figure 19, lorsque des composants accèdent à des pièces jointes référencées, les données des pièces jointes apparaissent sous forme de tableau octal.

Chaque pièce jointe référencée d'un message SOAP contient également un élément **attachments** correspondant dans l'objet SMO. L'élément **attachments** inclut le type de contenu de la pièce jointe et le chemin d'accès à l'élément de corps de message dans lequel la pièce jointe est conservée.

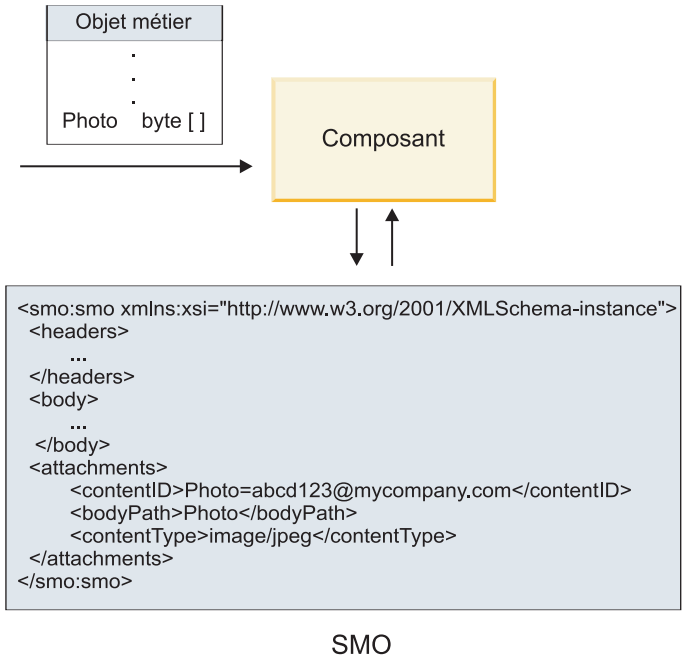


Figure 20. Affichage des pièces jointes référencées dans l'objet SMO

Important : Le chemin d'accès à l'élément de corps du message n'est pas automatiquement mis à jour si le message est converti et la pièce jointe, déplacée. Vous pouvez utiliser un flux de médiation pour mettre à jour l'élément **attachments** avec le nouveau chemin d'accès (par exemple, dans le cadre de la conversion ou à l'aide d'un configurateur d'élément de message distinct).

Mode de construction des messages SOAP sortant

Utilisez Integration Designer pour configurer une liaison d'importation de service Web (JAX-WS) de sorte qu'elle appelle un service Web externe. La liaison d'importation est configurée avec un document WSDL qui décrit le service Web à appeler et définit les portions du message qui doivent être transmises comme pièces jointes. Vous pouvez aussi préciser l'un des choix suivants sur la page Specify the WS-I AP 1.0 compliance (Spécifier la compatibilité WS-I AP 1.0) de Integration Designer :

- **Use WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP compatible WS-I AP 1.0)**

  Si vous sélectionnez cette option, indiquez aussi quelle partie du message doit être liée au corps SOAP ; toutes les autres sont liées à des pièces jointes ou à des en-têtes. Les messages envoyés par le lien n'inclut pas d'éléments du corps SOAP faisant référence aux pièces jointes ; la relation est exprimée au moyen de l'ID de contenu des pièces jointes incluant le nom de la partie du message.
- **Use non WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP non compatible WS-I AP 1.0)**

  Si vous sélectionnez cette option, qui est celle par défaut, la première partie du message est liée au corps SOAP ; toutes les autres sont liées aux pièces jointes et en-têtes. Les messages envoyés par la liaison incluent un ou plusieurs éléments du corps SOAP faisant référence aux pièces jointes au moyen d'un attribut **href**.

Remarque : La portion qui représente une pièce jointe, telle que définie dans le WSDL, doit être de type simple (base64Binary ou hexBinary). Si une portion est définie par un type complexe (complexType), celle-ci ne peut pas être liée comme pièce jointe.

Traitement sortant des pièces jointes référencées

La liaison d'importation utilise les informations dans le SMO pour déterminer comment les portions de messages binaires de niveau supérieur sont envoyées comme pièces jointes.

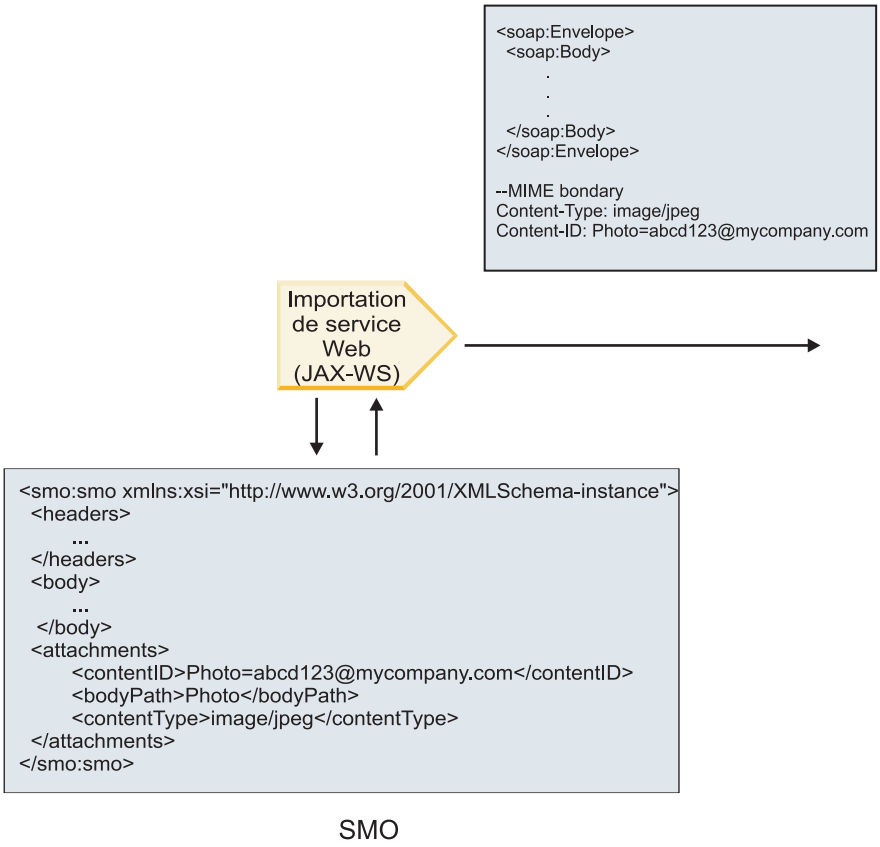


Figure 21. Mode d'accès à la pièce jointe référencée de l'objet SMO pour créer le message SOAP

L'élément **attachments** n'est présent dans l'objet SMO que si un composant de flux de médiation est connecté directement à l'importation ou l'exportation ; il n'est pas transmis entre les autres types de composant. Si les valeurs sont requises dans un module contenant d'autres types de composants, un composant de flux de médiation doit être utilisé pour copier ces valeurs dans un emplacement où elles sont accessibles dans le module et un autre composant de flux de médiation doit être utilisé pour définir les valeurs correctes avant un appel sortant par l'intermédiaire d'une importation de service Web.

La liaison utilise une combinaison des conditions suivantes pour déterminer comment (ou si) le message est envoyé :

- s'il existe une liaison MIME WSDL pour la portion du message binaire de niveau supérieur et, si tel est le cas, comment le type de contenu est défini,
- s'il existe un élément **attachments** dans le SMO dont la valeur **bodyPath** fait référence à une portion binaire de haut niveau,

Mode de création des pièces jointes quand un élément attachments se trouve dans le SMO.

Le tableau suivant montre comment une pièce jointe est créée et envoyée si le SMO contient un élément **attachments** avec un élément **bodyPath** correspondant une partie du nom du message :

Tableau 32. Mode de génération de la pièce jointe

Statut de la liaison MIME WSDL pour la portion de message binaire de haut niveau	**Mode de création et d'envoi du message**
Présent avec l'une des situations suivantes : • Aucun type de contenu défini pour la portion du message • Multiple types de contenu définis • Type de contenu avec caractère générique défini	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est définie à la valeur de l'élément attachments le cas échéant ; sinon, une valeur est générée. **Content-Type** est défini à la valeur de l'élément attachments le cas échéant ; sinon, il est défini à **application/octet-stream**.
Présent avec du contenu sans caractère générique pour la portion du message	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est définie à la valeur de l'élément attachments le cas échéant ; sinon, une valeur est générée. **Content-Type** est défini à la valeur de l'élément attachments le cas échéant ; sinon, il est défini au type défini dans l'élément de contenu MIME WSDL.
Non présent	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est définie à la valeur de l'élément attachments le cas échéant ; sinon, une valeur est générée. **Content-Type** est défini à la valeur de l'élément attachments le cas échéant ; sinon, il est défini à **application/octet-stream**. **Remarque :** L'envoi de portions de messages en tant que pièces jointes quand elles ne sont pas définies comme telles dans le langage WSDL peut empêcher la conformité avec WS-I Attachments Profile 1.0 et doit donc être si possible évité.

Mode de création des pièces jointes quand aucun élément attachments ne se trouve dans le SMO

Le tableau suivant montre comment une pièce jointe est créée et envoyée si le SMO ne contient pas d'élément **attachments** avec un élément **bodyPath** correspondant à une partie du nom du message :

Tableau 33. Mode de génération de la pièce jointe

Statut de la liaison MIME WSDL pour la portion de message binaire de haut niveau	**Mode de création et d'envoi du message**
Présent avec l'une des situations suivantes : • Aucun type de contenu défini pour la portion du message • Multiple types de contenu définis • Type de contenu avec caractère générique défini	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est généré. **Content-Type** est défini à **application/octet-stream**.

Tableau 33. Mode de génération de la pièce jointe (suite)

Statut de la liaison MIME WSDL pour la portion de message binaire de haut niveau	**Mode de création et d'envoi du message**
Présent avec du contenu sans caractère générique pour la portion du message	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est généré. **Content-Type** est défini au type défini dans l'élément de contenu MIME WSDL.
Non présent	La portion du Message n'est pas envoyée en tant que pièce jointe.

Important : Comme décrit dans la «représentation XML de l'objet SMO,» la primitive de médiation de mappage convertit les messages à l'aide d'une transformation XSLT 1.0. La transformation agit sur une sérialisation XML de l'objet SMO. La primitive de médiation de Mappage permet de spécifier la racine de la sérialisation et l'élément racine du document XML reflète cette racine.

Lorsque vous envoyez des messages SOAP comportant des pièces jointes, l'élément racine choisi détermine le mode de propagation des pièces jointes.

- Si vous utilisez «/body» comme racine de la mappe XML, toutes les pièces jointes sont propagées dans la mappe par défaut.
- Si vous utilisez «/» comme racine de la mappe, vous pouvez contrôler la propagation des pièces jointes.

Pièces jointes non référencées : Advanced

Vous pouvez envoyer et recevoir des pièces jointes *non référencées* qui ne sont pas déclarées dans l'interface de service.

Dans ce type de message, le corps SOAP constitue la première partie et les pièces jointes figurent dans les parties suivantes. Aucune référence à la pièce jointe n'est incluse dans le corps SOAP.

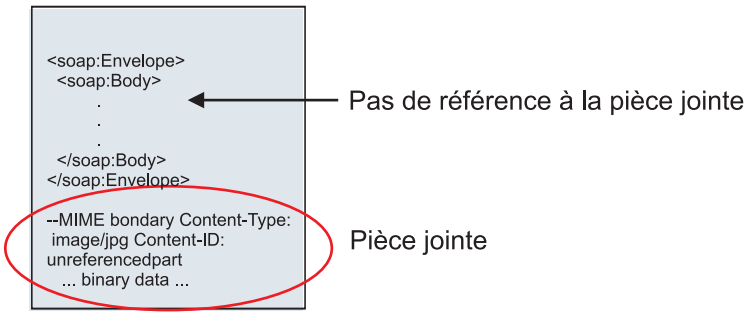


Figure 22. Message SOAP avec une pièce jointe non référencée

Vous pouvez envoyer un message SOAP avec une pièce jointe non référencée par une exportation de service Web vers une importation de service Web. Le message de sortie envoyé au service web cible contient la pièce jointe.

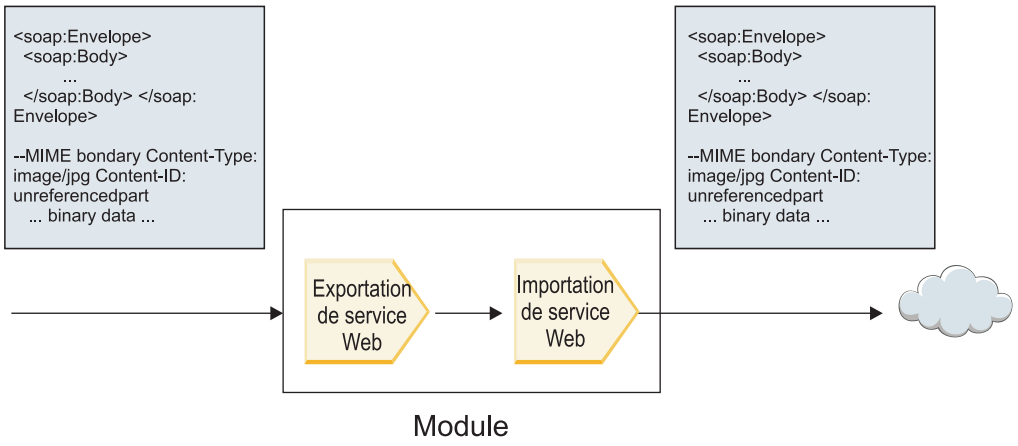


Figure 23. Pièce jointe transitant par un module SCA

Dans la figure 23, le message SOAP comprenant une pièce jointe transite par le module sans modification.

Vous pouvez également modifier le protocole SOAP à l'aide d'un composant de flux de médiation. Par exemple, vous pouvez utiliser le composant de flux de médiation pour extraire des données du message SOAP (dans ce cas, des données binaires dans le corps du message) et créer un message SOAP avec pièces jointes. Les données sont traitées comme partie de l'élément attachments d'un objet de message de service (SMO).

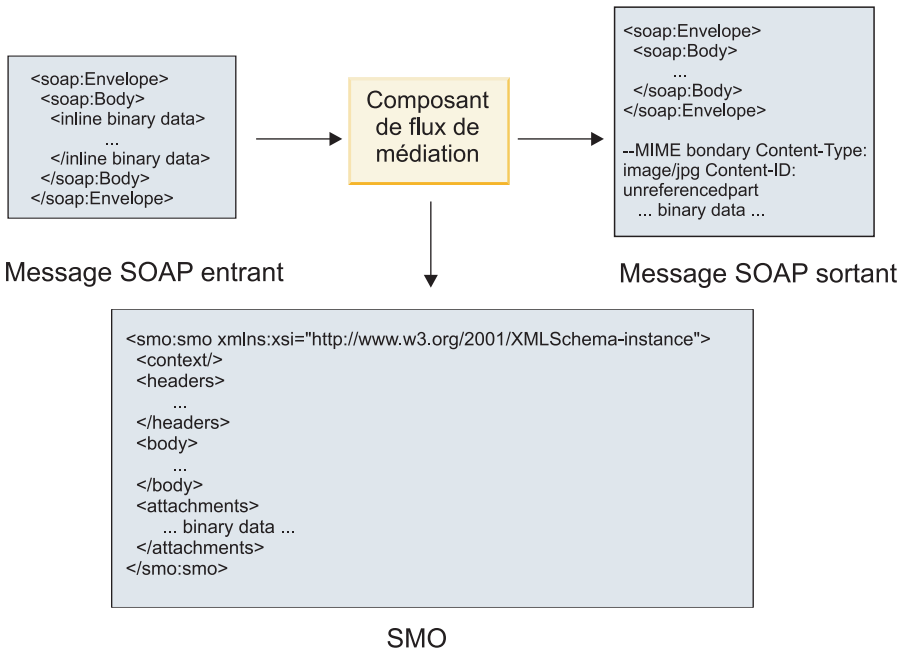


Figure 24. Message traité par un composant de flux de médiation

Inversement, le composant de flux de médiation peut transformer le message entrant en extrayant et en codant la pièce jointe, puis en transmettant le message sans pièce jointe.

Plutôt que d'extraire les données d'un message SOAP entrant pour former un message SOAP avec pièces jointes, vous pouvez obtenir les données de la pièce jointe à partir d'une source externe (par exemple, une base de données).

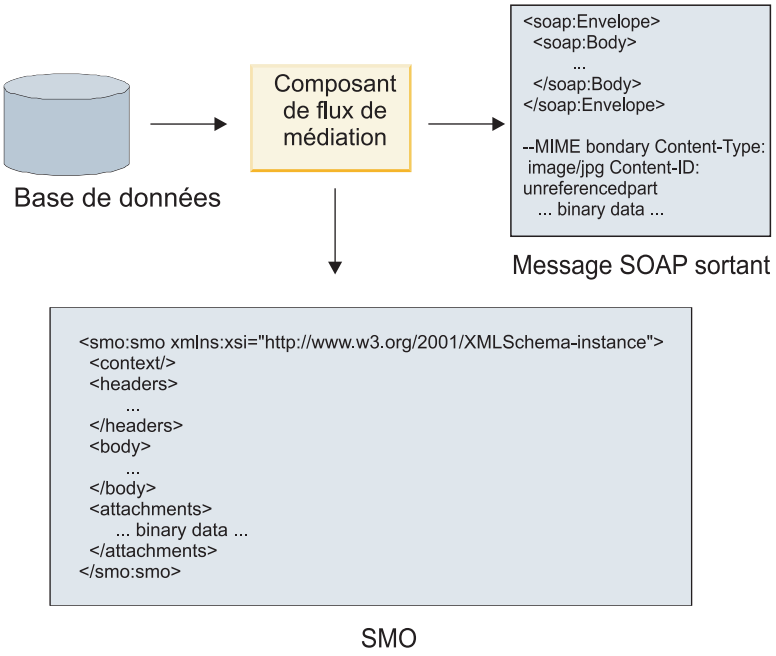


Figure 25. Pièce jointe obtenue d'une base de données et ajoutée au message SOAP

Inversement, le composant de flux de médiation peut extraire la pièce jointe d'un message SOAP entrant et traiter le message (par exemple, stocker la pièce jointe dans une base de données).

La propagation des pièces jointes non référencées n'est possible qu'entre composants de flux de médiation. Si un autre type de composant doit accéder à la pièce jointe ou servir de cible lors de la propagation de celle-ci, utilisez un composant de flux de médiation pour déplacer la pièce jointe vers un emplacement accessible par le composant.

Important : Comme décrit dans la «représentation XML de l'objet SMO,» la primitive de médiation de mappage convertit les messages à l'aide d'une transformation XSLT 1.0. La transformation agit sur une sérialisation XML de l'objet SMO. La primitive de médiation de Mappage permet de spécifier la racine de la sérialisation et l'élément racine du document XML reflète cette racine.

Lorsque vous envoyez des messages SOAP comportant des pièces jointes, l'élément racine choisi détermine le mode de propagation des pièces jointes.

- Si vous utilisez «/body» comme racine de la mappe XML, toutes les pièces jointes sont propagées dans la mappe par défaut.
- Si vous utilisez «/» comme racine de la mappe, vous pouvez contrôler la propagation des pièces jointes.

Utilisation d'une liaison de style Document WSDL avec des messages composites : Advanced

L'organisation WS-I (Web Services Interoperability Organization) a défini un ensemble de règles sur la manière dont les services web doivent être décrits par l'intermédiaire d'un WSDL et dont les messages SOAP correspondants doivent être constitués pour assurer leur interopérabilité.

Ces règles sont spécifiées dans WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). En particulier, WS-I Basic Profile 1.1 R2712 indique : "Une liaison document/littéral DOIT être mise en série comme une ENVELOPPE avec un soap:Body dont l'élément enfant est une instance de la déclaration de l'élément global référencé par la portion wsdl:message correspondante."

Cela signifie que si une liaison SOAP de style document est utilisée pour une opération dont les messages (entrée, sortie ou incident) sont définis avec plusieurs portions, une seule de ces portions doit être associée au corps du message SOAP pour la compatibilité à WS-I Basic Profile 1.1.

En outre, WS-I Attachments Profile 1.0 R2941 précise : "Un wsdl:binding d'une DESCRIPTION DOIT lier tous les wsdl:part d'un wsdl:message dans le wsdl:portType auquel il fait référence à l'un des suivants : soapbind:body, soapbind:header, soapbind:fault , soapbind:headerfault ou mime:content.".

Cela signifie que si une liaison SOAP de style document est utilisée pour une opération dont les messages (entrée, sortie ou incident) sont définis avec plusieurs portions, toutes les portions autres que celle sélectionnée pour être liée au corps SOAP doivent être liées en tant que pièces jointes ou en-têtes.

L'approche suivante est utilisée lorsque des descriptions WSDL sont générées pour les exportations avec des liaisons de service Web (JAX-WS et JAX-RPC) dans le cas suivant :

- Vous pouvez choisir quelle portion du message est liée au corps SOAP s'il existe plusieurs éléments typés non binaire. S'il n'existe qu'un seul élément typé non binaire, cet élément est lié automatiquement au corps SOAP.
- Pour la liaison JAX-WS, toutes les autres portions de message de type "hexBinary" ou "base64Binary" sont associées comme pièces jointes référencées. Voir «Pièces jointes référencées : composants de message de niveau supérieur», à la page 89.
- Toutes les autres portions de message sont associées comme en-têtes SOAP.

Les liaisons d'importation JAX-RPC et JAX-WS traitent la liaison SOAP d'un document WSDL existant avec des messages de style document en plusieurs parties, même si elle n'associe pas plusieurs portions au corps du message SOAP ; toutefois, vous ne pouvez pas générer de clients de service Web pour ces documents WSDL dans Rational Application Developer.

Remarque : La liaison JAX-RPC ne prend pas en charge les pièces jointes.

Le modèle recommandé lors de l'utilisation de messages composites avec une opération dont le style de document est une liaison SOAP est donc le suivant :

1. Usez le style encapsulé dans un document/littéral. Dans ce cas, les messages comportent toujours une seule partie, mais le référencement des messages doit être supprimé (comme décrit dans «Pièces jointes non référencées», à la page 94) ou être de type swaRef (comme décrit dans «Pièces jointes référencées : éléments de type swaRef», à la page 85).
2. Utilisé le style RPC/littéral. Dans ce cas, il n'existe aucune restriction sur la liaison WSDL en termes de nombre de portions associées au corps du message SOAP ; le message SOAP résultant possède toujours un enfant unique qui représente l'opération appelée, avec les portions de message correspondant aux enfants de cet élément.
3. Pour la liaison JAX-WS, il doit exister au plus une portion de message qui ne soit pas du "hexBinary" ou "base64Binary", sauf s'il est acceptable de lier les autres portions non binaires à des en-têtes SOAP.
4. Les autres cas ont le comportement décrit.

Remarque : Il existe des restrictions supplémentaires si vous utilisez des messages SOAP ne respectant pas le WS-I *Basic Profile Version 1.1*.

- La première portion du message doit être non binaire.
- Lorsqu'elle reçoit des messages SOAP de style document composite avec des pièces jointes référencées, la liaison JAX-WS s'attend à ce que chaque pièce jointe référencée soit représentée par un élément enfant de corps de message SOAP avec un attribut href dont la valeur identifie la pièce jointe d'après son ID contenu. La liaison JAX-WS envoie les pièces jointes référencées de tels messages de la même manière. Ce comportement n'est pas compatible avec WS-I Basic Profile.

Pour que vos messages soient compatibles avec Basic Profile, suivez l'approche 1, à la page 97 ou 2, à la page 97 de la liste précédente ou évitez d'utiliser des pièces jointes référencées pour de tels messages et utilisez à la place des pièces jointes non référencées ou de type swaRef.

Liaisons HTTP :

La liaison HTTP permet de relier une architecture SOA à HTTP. Par conséquent, les applications HTTP existantes ou récemment développées peuvent être intégrées aux environnements d'architecture SOA (Service Oriented Architecture).

Le protocole HTTP (Hypertext Transfer Protocol) est largement utilisé pour le transfert d'informations sur le web. Lorsque vous travaillez avec une application externe qui utilise le protocole HTTP, une liaison HTTP est nécessaire. La liaison HTTP transforme les données entrantes en tant que message au format natif en un objet métier dans une application SCA. La liaison HTTP peut également transformer les données sortantes en tant qu'objet métier au format natif attendu par l'application externe.

Remarque : Si vous voulez interagir avec des clients et des services qui utilisent le protocole SOAP/HTTP de services Web, vous pouvez utiliser l'une des liaisons de service Web qui fournissent des fonctionnalités supplémentaires quant à la gestion des qualités de service standard des services Web.

La liste suivante répertorie certains scénarios courants pour l'utilisation de la liaison HTTP :

- Les services hébergés sur SCA peuvent appeler des applications HTTP en utilisant une importation HTTP.
- Les services hébergés sur SCA peuvent s'afficher eux-mêmes en tant qu'applications conformes à HTTP pour pouvoir être utilisés par des clients HTTP à l'aide d'une exportation HTTP.
- IBM Business Process Manager et Process Server peuvent communiquer entre eux via une infrastructure HTTP ; les utilisateurs peuvent donc gérer leurs communications en fonction de normes d'entreprise.
- IBM Business Process Manager et Process Server peuvent agir comme médiateurs des communications HTTP en tranformant et en routant les messages, ce qui améliore l'intégration des applications à l'aide d'un réseau HTTP.
- IBM Business Process Manager et Process Server peuvent être utilisés pour établir un pont entre HTTP et d'autres protocoles, comme les services Web SOAP/HTTP, les adaptateurs de ressources basés sur JCA (Java Connector Architecture), JMS, etc.

Pour plus d'informations sur la création de liaisons d'exportation et d'importation HTTP, consultez le centre de documentation de Integration Designer. Voir les rubriques**Développement des applications d'intégration** > **Accès aux services externes avec HTTP**>.

Présentation des liaisons HTTP :

La liaison HTTP offre une connectivité aux applications hébergées sur HTTP. Elle sert d'intermédiaire aux communications entre les applications HTTP et permet aux applications existantes basées sur HTTP d'être appelées depuis un module.

Liaisons d'importation HTTP

La liaison d'importation HTTP offre une connectivité sortante depuis les applications SCA (Service Component Architecture) vers des applications ou un serveur HTTP.

L'importation appelle une adresse URL de noeud final HTTP. L'URl peut être spécifiée de l'une des trois façons suivantes :

- L'URL peut être définie de manière dynamique dans les en-têtes HTTP par l'intermdiaire d'une URL de substitution dynamique.

- L'URL peut être définie de manière dynamique dans l'élément d'adresse cible SMO.
- L'URL peut être spécifiée en tant que propriété de configuration sur l'importation.

Cet appel est toujours synchrone par nature.

Bien que les appels HTTP soient toujours de type demande-réponse, l'importation HTTP prend en charge à la fois les opérations unidirectionnelles et bidirectionnelles et ignore la réponse dans le cas d'une opération unidirectionnelle.

Liaisons d'exportation HTTP

La liaison d'exportation HTTP offre une connectivité entrante depuis les applications HTTP vers une application SCA.

Une adresse URL est définie sur l'exportation HTTP. Les applications HTTP qui veulent envoyer des messages de demande à l'exportation utilisent cette adresse URL pour appeler l'exportation.

L'exportation HTTP prend également en charge les commandes ping.

Liaisons HTTP lors de l'exécution

Une importation avec une liaison HTTP au moment de l'exécution envoie une demande avec ou sans données dans le corps du message à partir de l'application SCA vers le service Web externe. Cette demande est émise à partir de l'application SCA vers le service Web externe, comme indiqué dans figure 26.

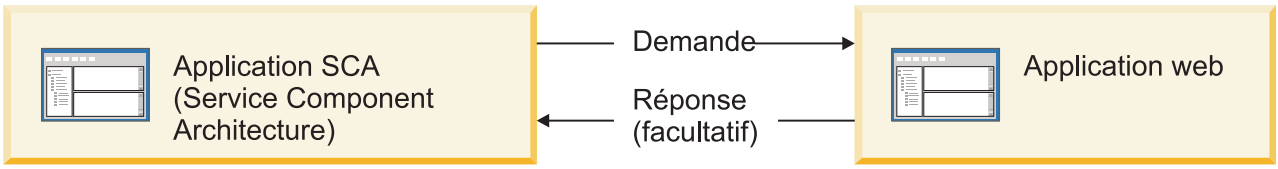


Figure 26. Flux d'une demande à partir de l'application SCA vers une application Web

L'importation avec la liaison HTTP peut éventuellement recevoir des données en retour issues d'une application Web dans la réponse à une demande.

Avec une exportation, la demande est faite par une application client à un service Web, comme indiqué dans figure 27.

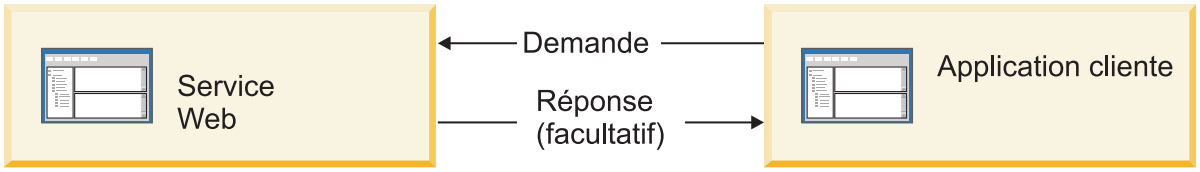


Figure 27. Flux d'une demande depuis l'application client vers le service Web.

Le service web est une application Web exécutée sur le serveur. L'exportation est implémentée dans cette application web en tant que servlet de telle sorte que le client envoie sa requête à une adresse URL. Le servlet transmet la requête à l'application SCA dans l'environnement d'exécution.

L'exportation peut éventuellement envoyer des données à l'application client en réponse à la requête.

En-têtes HTTP :

Les liaisons d'importation et d'exportation HTTP permettent d'effectuer la configuration des en-têtes HTTP et de leurs valeurs pour les messages sortants. L'importation HTTP utilise ces en-têtes pour les requêtes, tandis que l'exportation HTTP les exploite pour les réponses.

Les en-têtes et données de contrôle configurés de façon statique ont priorité sur les valeurs définies dynamiquement au moment de l'exécution. Toutefois, les valeurs de substitution dynamique d'adresse URL, de version et de méthode remplacement les valeurs statiques, qui sont dans les autres cas considérées comme valeurs par défaut.

La liaison prend en charge la nature dynamique de l'URL d'importation HTTP en déterminant la valeur des Méthode, Version et URL cible HTTP en phase d'exécution. Ces valeurs sont déterminées par l'extraction des valeurs de Référence de point de contact, URL de substitution dynamique, Version et Méthode.

- Pour Référence de point de contact, utilisez les API com.ibm.websphere.sca.addressing.EndpointReference ou définissez la zone /headers/SMOHeader/Target/address dans l'en-tête SMO.
- Pour URL de substitution dynamique, Version et Méthode, utilisez la section des paramètres de contrôle HTTP du message SCA (Service Component Architecture). Notez que l'URL de substitution dynamique est prioritaire par rapport à la référence de point de contact cible, mais que cette dernière s'appliquant entre les liaisons, elle est préférable et doit être utilisée chaque fois que possible.

Les données de contrôle et d'en-tête contenues dans les messages sortants sous les liaisons d'importation et d'exportation HTTP sont traitées dans l'ordre suivant :

1. Informations de contrôle et d'en-tête excluant les valeurs de méthode, de version et d'URL de substitution dynamique HTTP du message SCA (priorité la plus faible)
2. Les modifications effectuées au niveau de l'exportation/importation via la console d'administration
3. Les modifications effectuées au niveau de la méthode d'exportation ou d'importation via la console d'administration
4. Adresse cible spécifiée par l'intermédiaire de la référence de point de contact ou de l'en-tête SMO
5. URL de substitution dynamique, Version et Méthode du message SCA
6. Informations de contrôle et d'en-tête provenant du gestionnaire de données ou de la liaison de données (priorité supérieure)

L'importation et l'exportation HTTP ne rempliront les paramètres de contrôle et les en-têtes de direction sortants avec les données provenant du message entrant (HTTPExportRequest et HTTPImportResponse) que si la propagation de l'en-tête de protocole a la valeur **True**. Inversement, l'exportation et l'importation HTTP ne liront et ne traiteront les paramètres de contrôle et en-têtes sortants (HTTPExportResponse et HTTPImportRequest) que si la propagation de l'en-tête de protocole a la valeur **True**.

Remarque : Les modifications de la liaison de données ou du gestionnaire de données apportées aux en-têtes ou aux paramètres de contrôle dans la réponse d'importation ou la requête d'exportation n'altéreront pas les instructions de traitement du message au sein de la liaison d'imortation ou d'exportation et ne doivent être utilisées que pour diffuser les valeurs modifiées vers les composants SCA en aval.

Le service de contexte est chargé de la propagation du contexte (y compris les en-têtes de protocole comme l'en-tête HTTP et le contexte utilisateur comme l'ID de compte) tout au long d'un chemin d'appel SCA. Lors du développement dans IBM Integration Designer, vous pouvez contrôler la propagation du contexte par l'intermédiaire des propriétés d'importation et d'exportation. Pour plus de détails, reportez-vous aux informations relatives aux liaisons d'importation et d'exportation dans le centre de documentation IBM Integration Designer.

Structures d'en-tête HTTP fournies et prise en charge

Le tableau 34 spécifie en détail les demandes de requête et de réponse d'importation HTTP et d'exportation HTTP.

Tableau 34. Informations d'en-tête HTTP fournies

Nom du contrôle	Demande d'importation HTTP	Réponse d'importation HTTP	Demande d'exportation HTTP	Réponse d'exportation HTTP
URL	Ignored	Non définie	Lecture à partir du message de demande. **Remarque :** La chaîne de requête est également incluse dans le paramètre de contrôle de l'URL.	Ignored
Version (valeurs possibles : 1.0, 1.1 - Par défaut : 1.1)	Ignored	Non définie	Lecture à partir du message de demande	Ignored
Méthode	Ignored	Non définie	Lecture à partir du message de demande	Ignored
URL de substitution dynamique	Si elle est définie dans le gestionnaire de données ou la liaison de données, elle remplace l'URL d'importation HTTP. Inscrite sur le message dans la ligne de requête. **Remarque :** La chaîne de requête est également incluse dans le paramètre de contrôle de l'URL.	Non définie	Non définie	Ignored
Version de substitution dynamique	Si elle est définie, elle se substitue à la version d'importation HTTP. Elle est inscrite sur le message dans la ligne de requête.	Non définie	Non définie	Ignored
Méthode de substitution dynamique	Si elle est définie, elle se substitue à la méthode d'importation HTTP. Elle est inscrite sur le message dans la ligne de requête.	Non définie	Non définie	Ignored

Tableau 34. Informations d'en-tête HTTP fournies (suite)

Nom du contrôle	Demande d'importation HTTP	Réponse d'importation HTTP	Demande d'exportation HTTP	Réponse d'exportation HTTP
Type de support (ce paramètre de contrôle transport une partie de la valeur de l'en-tête HTTP de type de contenu.)	Si elle existe, elle est inscrite dans le message en tant que partie de l'en-tête de type de contenu. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par le gestionnaire de données ou la liaison de données.	Lecture à partir du message de réponse, en-tête Content-Type	Lecture à partir du message de demande, en-tête Content-Type	Si elle existe, elle est inscrite dans le message en tant que partie de l'en-tête Content-Type. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par le gestionnaire de données ou la liaison de données.
Jeu de caractères (par défaut : UTF-8)	Si elle existe, elle est inscrite dans le message en tant que partie de l'en-tête de type de contenu. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par la liaison de données.	Lecture à partir du message de réponse, en-tête Content-Type	Lecture à partir du message de demande, en-tête Content-Type	Pris en charge, inscrit dans le message en tant que partie de l'en-tête de type de contenu. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par la liaison de données.
Codage de transfert (valeurs possibles : chunked, identity. Valeur par défaut: identity)	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage appliqué lors de la transformation du message.	Lecture à partir du message de réponse	Lecture à partir du message de demande	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage appliqué lors de la transformation du message.
Codage de contenu (valeurs possibles : gzip, x-gzip, deflate, identity. Valeur par défaut: identity)	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage de la charge.	Lecture à partir du message de réponse	Lecture à partir du message de demande	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage de la charge.
Longueur de contenu	Ignorée	Lecture à partir du message de réponse	Lecture à partir du message de demande	Ignorée
StatusCode (par défaut : 200)	Non prise en charge.	Lecture à partir du message de réponse	Non prise en charge.	Si elle existe, elle est inscrite dans le message dans la ligne de réponse
ReasonPhrase (par défaut : OK)	Non prise en charge.	Lecture à partir du message de réponse	Non prise en charge.	Valeur de contrôle ignorée. La valeur contenue sur la ligne de réponse du message est générée à partir de StatusCode.

Tableau 34. Informations d'en-tête HTTP fournies (suite)

Nom du contrôle	Demande d'importation HTTP	Réponse d'importation HTTP	Demande d'exportation HTTP	Réponse d'exportation HTTP
Authentification (propriétés multiples)	Si elle existe, elle est utilisée pour établir l'en-tête d'authentification de base. **Remarque :** La valeur de cet en-tête est uniquement encodée dans le protocole HTTP. Dans l'architecture SCA, cette donnée est décodée et transmise sous forme de texte en clair.	Non disponible	Lecture à partir de l'en-tête d'authentification de base du message de demande. La présence de cet en-tête n'indique pas que l'utilisateur a été authentifié. Il convient de contrôler l'authentification via la configuration du servlet. **Remarque :** La valeur de cet en-tête est uniquement encodée dans le protocole HTTP. Dans l'architecture SCA, cette donnée est décodée et transmise sous forme de texte en clair.	Non disponible
Proxy (contient des propriétés multiples : Host, Port, Authentication)	Si elle existe, elle permet d'établir la connexion via le serveur Proxy.	Non disponible	Non disponible	Non disponible
SSL (contient des propriétés multiples : Keystore, Keystore Password, Trustore, Trustore Password, ClientAuth)	Si elle est remplie et que L'URL de destination est HTTPS, elle est utilisée pour établir une connexion via SSL.	Non disponible	Non disponible	Non disponible

Liaisons de données HTTP :

Pour chaque mappage de données distinct entre un message SCA (Service Component Architecture) et un message de protocole HTTP, un gestionnaire de données ou une liaison de données HTTP doit être configuré. Les gestionnaires de données, qui fournissent une interface indépendante des liaisons pouvant être réutilisée avec différentes liaisons de transport, représentent l'approche recommandée ; les liaisons de données sont spécifiques à une liaison de transport particulière. Des classes de liaisons de données propres à HTTP sont fournies ; vous pouvez également créer des liaisons ou des gestionnaires de données personnalisés.

Remarque : Les trois classes de liaison de données HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML et HTTPServiceGatewayDataBinding) sont dépréciées à partir de IBM Business Process Manager, version 7.0. Au lieu d'utiliser les liaisons de données décrites dans cette rubrique, prenez en compte les gestionnaires de données suivants :

- Utilisez SOAPDataHandler au lieu de HTTPStreamDataBindingSOAP.
- Utilisez UTF8XMLDataHandler au lieu de HTTPStreamDataBindingXML
- Utilisez GatewayTextDataHandler au lieu de HTTPServiceGatewayDataBinding

Les liaisons de données suivantes sont disponibles pour les importations et les exportations HTTP : liaison de données binaire, liaison de données XML et liaison de données SOAP. Une liaison de données de réponse n'est pas nécessaire pour les opérations unidirectionnelles. Une liaison de données est représentée par le nom d'une classe Java dont les instances permettent à la fois les conversions de HTTP vers ServiceDataObject et vice versa. Un sélecteur de fonction doit être utilisé lors d'une exportation pour déterminer, conjointement avec les liaisons de méthodes, la liaison de données utilisée et l'opération appelée. Les liaisons de données fournies sont les suivantes :

- Les liaisons de données binaires qui traitent le corps des messages comme des données binaires non structurées. L'implémentation du schéma XSD de liaisons de données binaires se présente comme suit :

```
<xsd:schema elementFormDefault="qualified"
  targetNamespace="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:tns="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:complexType name="HTTPBaseBody">
    <xsd:sequence/>
  </xsd:complexType>

  <xsd:complexType name="HTTPBytesBody">
    <xsd:complexContent>
      <xsd:extension base="tns:HTTPBaseBody">
        <xsd:sequence>
          <xsd:element name="value" type="xsd:hexBinary"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

- Les liaisons de données XML qui prennent en charge le corps des messages en tant que données XML. L'implémentation des liaisons de données XML est similaire à celle des liaisons de données XML JMS et n'impose aucune restriction quant au schéma d'interface.
- Les liaisons de données SOAP qui prennent en charge le corps des messages en tant que données SOAP. L'implémentation des liaisons de données SOAP n'impose aucune restriction quant au schéma d'interface.

Implémentation des liaisons de données HTTP personnalisées

Cette section indique comment implémenter une liaison de données HTTP personnalisée.

Remarque : L'approche recommandée consiste à implémenter un gestionnaire de données personnalisé car il peut être réutilisé avec différentes liaisons de transport.

HTTPStreamDataBinding est la principale interface de gestion des messages HTTP personnalisés. Cette interface est conçue pour permettre le traitement de charges utiles importantes. Toutefois, pour que ce type d'implémentation fonctionne, cette liaison de données doit renvoyer les informations de contrôle et les en-têtes avant que le message ne soit inséré dans le flux.

Les méthodes et leur ordre d'exécution (indiqué ci-dessous) doivent être implémentés par la liaison de données personnalisée.

Pour personnaliser une liaison de données, créez une classe implémentant HTTPStreamDataBinding. La liaison de données doit comporter les quatre propriétés privées suivantes :

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders
- private yourNativeDataType nativeData

La liaison HTTP appellera la liaison de données personnalisée dans l'ordre suivant :

- Traitement sortant (objet de données vers format natif) :
  1. setDataObject(...)
  2. setHeaders(...)
  3. setControlParameters(...)
  4. setBusinessException(...)
  5. convertToNativeData()
  6. getControlParameters()
  7. getHeaders()
  8. write(...)
- Traitement entrant (format natif vers objet de données) :
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Vous devez appeler setDataObject(...) dans convertFromNativeData(...) pour définir la valeur de dataObject, qui est convertie du format de données natif en propriété privée "pDataObject".

```
public void setDataObject(DataObject dataObject)
   throws DataBindingException {
  pDataObject = dataObject;

}
public void setControlParameters(HTTPControl arg0) {
  this.pCtrl = arg0;
}

public void setHeaders(HTTPHeaders arg0) {
  this.pHeaders = arg0;
}
/*
* Ajouter l'en-tête http "IsBusinessException" dans pHeaders.
* Deux étapes :
* 1. Supprimer d'abord tous les en-têtes portant le nom IsBusinessException (insensible à la casse).
*   Cette opération garantit qu'un seul en-tête est présent.
* 2. Ajouter le nouvel en-tête "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //Supprimer d'abord tous les en-têtes portant le nom IsBusinessException (insensible à la casse).
  //Cette opération garantit qu'un seul en-tête est présent.
  //Ajouter le nouvel en-tête "IsBusinessException", exemple de code :
  HTTPHeader header=HeadersFactory.eINSTANCE.createHTTPHeader();
  header.setName("IsBusinessException");
  header.setValue(Boolean.toString(isBusinessException));
  this.pHeaders.getHeader().add(header);

}
public HTTPControl getControlParameters() {
  return pCtrl;
}
public HTTPHeaders getHeaders() {
  return pHeaders;
}
public DataObject getDataObject() throws DataBindingException {
  return pDataObject;
}
```

```
/*
* Extraire l'en-tête "IsBusinessException" de pHeaders, renvoyer sa valeur booléenne
*/
public boolean isBusinessException() {
  String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
  boolean result=Boolean.parseBoolean(headerValue);
  return result;
}
public void convertToNativeData() throws DataBindingException {
  DataObject dataObject = getDataObject();
  this.nativeData=realConvertWorkFromSDOToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
  //Méthode développée par le client pour
  //Lire des données de HTTPInputStream
  //Les convertir en objet de données (DataObject)
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

Liaisons EJB :

Les liaisons d'importation EJB (Enterprise JavaBeans permettent aux composants SCA d'appeler les services fournis par la logique métier de Java EE exécutée sur un serveur Java EE. Les liaisons d'exportation EJB permettent aux composants d'être affichés comme des EJB (Enterprise JavaBeans) pour que la logique métier de Java EE puisse appeler les composants SCA qui ne seraient autrement pas disponibles.

Liaisons d'importation EJB :

Les liaisons d'importation EJB permettent à un module SCA d'appeler des implémentations EJB en indiquant la manière dont le module utilisateur est lié à l'EJB externe. L'importation de services à partir d'une implémentation EJB externe permet aux utilisateurs de relier leur logique métier à l'environnement IBM Business Process Manager et de participer à un processus métier.

Vous pouvez utiliser Integration Designer pour créer des liaisons d'importation EJB. Vous pouvez utiliser l'une des procédures suivantes pour générer les liaisons :

- Création d'une importation EJB à l'aide de l'assistant de service externe

  Vous pouvez utiliser l'assistant de service externe de Integration Designer pour générer une importation EJB à partir d'une implémentation existante. L'assistant de service externe crée des services en fonction des critères que vous fournissez. Il génère ensuite des objets métier, des interfaces et des fichiers d'importation à partir des services détectés.
- Création d'une importation EJB à l'aide de l'éditeur d'assemblage

  Vous pouvez créer une importation EJB dans un diagramme d'assemblage à l'aide de l'éditeur d'assemblage de Integration Designer. A partir de la palette, vous pouvez utiliser une importation ou utiliser une classe Java pour créer la liaison EJB.

L'importation générée contient des liaisons de données qui établissent la connexion Java-WSDL, ce qui évite le recours à un composant de passerelle Java. Vous pouvez connecter directement un composant avec une référence WSDL (Web Services Description Language) à l'importation EJB qui communique avec un service EJB à l'aide d'une interface Java.

L'importation EJB peut interagir avec la logique métier Java EE à l'aide du modèle de programmation EJB 2.1 ou du modèle de programmation EJB 3.0.

L'appel de la logique métier Java EE peut être local (uniquement pour EJB 3.0) ou éloigné.

- L'appel local est utilisé lorsque vous souhaitez appeler la logique métier Java EE qui se trouve sur le même serveur que l'importation.

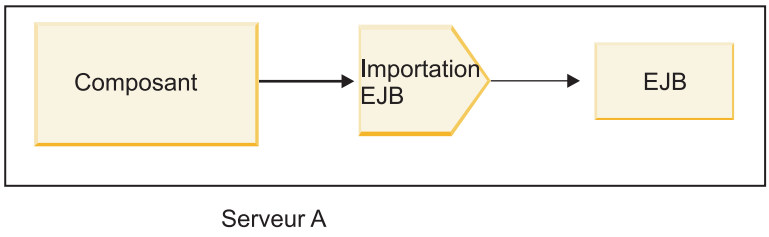


Figure 28. Appel en local d'un EJB (EJB 3.0 uniquement)

- L'appel éloigné est utilisé lorsque vous souhaitez appeler la logique métier Java EE qui ne se trouve pas sur le même serveur que l'importation.

  Par exemple, dans la figure ci-après, une importation EJB utilise RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) pour appeler une méthode EJB sur un autre serveur.

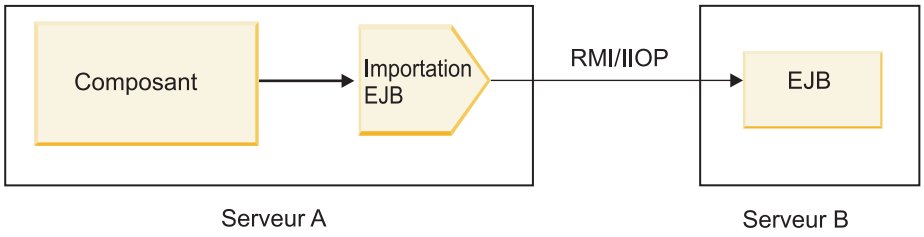


Figure 29. Appel éloigné d'un EJB

Lorsqu'il configure la liaison EJB, Integration Designer utilise le nom JNDI pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné).

Les liaisons d'importation EJB contiennent les principaux composants suivants :

- Gestionnaire de données JAX-WS
- Sélecteur d'erreurs EJB
- Sélecteur de fonction d'importation EJB

Si votre scénario utilisateur ne repose pas sur le mappage JAX-WS, vous risquez d'avoir besoin d'un gestionnaire de données, d'un sélecteur de fonction et d'un sélecteur d'erreur personnalisés pour accomplir les tâches habituellement dévolues aux composants des liaisons d'importation EJB. Ces tâches incluent le mappage normalement effectué par l'algorithme de mappage personnalisé.

Liaisons d'exportation EJB :

Les applications Java EE externes peuvent appeler un composant SCA par l'intermédiaire d'une liaison d'exportation EJB. L'utilisation d'une exportation EJB permet d'exposer des composants SCA pour que les applications Java EE externes puissent appeler ces composants à l'aide du modèle de programmation des EJB.

Remarque : L'exportation EJB est un bean sans état.

Vous pouvez utiliser Integration Designer pour créer des liaisons EJB. Vous pouvez utiliser l'une des procédures suivantes pour générer les liaisons :

- Création de liaisons d'exportation EJB à l'aide de l'assistant de service externe

Vous pouvez utiliser l'assistant de service externe de Integration Designer pour générer un service d'exportation EJB à partir d'une implémentation existante. L'assistant de service externe crée des services en fonction des critères que vous fournissez. Il génère ensuite des objets métier, des interfaces et des fichiers d'exportation à partir des services détectés.

- Création de liaisons d'exportation EJB à l'aide de l'éditeur d'assemblage.

  Vous pouvez créer une exportation EJB à l'aide de l'éditeur d'assemblage de Integration Designer.

Important : Un client J2SE (Java 2 Platform, Standard Edition) ne peut pas appeler le client d'exportation EJB qui est généré dans Integration Designer.

Vous pouvez générer la liaison à partir d'un composant SCA existant ou générer une exportation avec une liaison EJB pour une interface Java.

- Lorsque vous générez une exportation pour un composant SCA existant qui possède une interface WSDL, une interface Java est affectée à l'exportation.
- Lorsque vous générez une exportation pour une interface Java, vous pouvez sélectionner un WSDL ou une interface Java pour l'exportation.

  Remarque : Une interface Java utilisée pour créer une exportation EJB est soumise aux limitations suivantes en ce qui concerne les objets (exceptions et paramètres en entrée et en sortie) transmis comme paramètres sur un appel éloigné :
  - Ils doivent être de type concret (au lieu d'une interface ou d'un type abstrait).
  - Ils doivent respecter la spécification Enterprise JavaBeans. Ils doivent être sérialisables et contenir le constructeur no-argument par défaut et toutes les propriétés doivent être accessibles par l'intermédiaire de méthodes getter et setter.

    Pour obtenir des informations sur la spécification Enterprise JavaBeans, reportez-vous au site Web de Sun Microsystems, Inc. à l'adresse suivante : http://java.sun.com.

  En outre, l'exception doit correspondre à une exception vérifiée, héritée de java.lang.Exception et doit être unique (la génération de plusieurs exceptions de type vérifiée n'est pas prise en charge).

  Notez par ailleurs que l'interface métier d'un bean enterprise Java est une interface Java standard et ne doit pas étendre javax.ejb.EJBObject ou javax.ejb.EJBLocalObject. Les méthodes de l'interface métier ne doivent pas générer d'exception java.rmi.Remote.Exception.

Les liaisons d'exportation EJB peuvent interagir avec la logique métier Java EE à l'aide du modèle de programmation EJB 2.1 ou du modèle de programmation EJB 3.0.

L'appel peut être local (uniquement pour EJB 3.0) ou éloigné.

- L'appel local est utilisé lorsque la logique métier Java EE appelle un composant SCA qui se trouve sur le même serveur que l'exportation.
- L'appel éloigné est utilisé lorsque la logique métier Java EE se trouve pas sur le même serveur que l'exportation.

  Par exemple, dans la figure ci-après, un EJB utilise RMI/IIOP pour appeler un composant SCA sur un autre serveur.

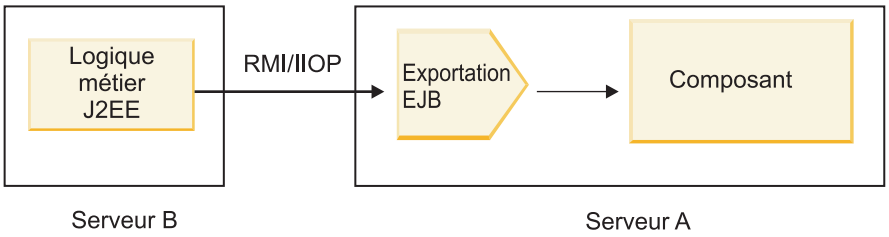


Figure 30. Appel éloigné d'un client vers un composant SCA par l'intermédiaire d'une exportation EJB

Lorsqu'il configure la liaison EJB, Integration Designer utilise le nom JNDI pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné).

Les liaisons d'exportation EJB contiennent les principaux composants suivants :

- Gestionnaire de données JAX-WS
- Sélecteur de fonction d'exportation EJB

Si votre scénario utilisateur ne repose pas sur le mappage JAX-WS, vous risquez d'avoir besoin d'un gestionnaire de données et d'un sélecteur de fonction personnalisés pour accomplir les tâches habituellement dévolues aux composants des liaisons d'exportation EJB. Ces tâches incluent le mappage normalement effectué par l'algorithme de mappage personnalisé.

Propriétés des liaisons EJB :

Les liaisons d'importation EJB utilisent leurs noms JNDI configurés pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné). Les liaisons d'importation et d'exportation EJB utilisent le gestionnaire de données JAX-WS pour la transformation des données. La liaison d'importation EJB utilise un sélecteur de fonction d'importation EJB et un sélecteur d'erreur EJB, tandis que la liaison d'exportation EJB utilise un sélecteur de fonction d'exportation EJB.

Noms JNDI et liaisons d'importation EJB :

Lorsqu'il configure la liaison EJB lors d'une importation, Integration Designer utilise le nom JNDI pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné).

Si aucun nom JNDI n'est spécifié, la liaison d'interface EJB est utilisée. Les noms par défaut créés varient suivant que vous appelez EJB 2.1 JavaBeans ou EJB 3.0 JavaBeans.

Remarque : Pour plus de détails sur les conventions de d'affectation de nom, reportez-vous à la rubrique "Présentation des liaisons d'application EJB 3.0" dans le centre de documentation WebSphere Application Server.

- EJB 2.1 JavaBeans

  Le nom JNDI par défaut présélectionné par Integration Designer correspond à la liaison EJB 2.1 par défaut, qui prend la forme **ejb/** plus l'interface home, séparés par des barres obliques.

  Par exemple, pour l'interface home de JavaBeans EJB 2.1 de com.mycompany.myremotebusinesshome, la liaison par défaut est :

  ```
  ejb/com/mycompany/myremotebusinesshome
  ```

  Pour EJB 2.1, seul l'appel EJB éloigné est pris en charge.
- EJB 3.0 JavaBeans

  Le nom JNDI par défaut présélectionné par Integration Designer pour le JNDI local correspond au nom de classe complet de l'interface locale précédé d'**ejblocal:**. Par exemple, pour l'interface complète de l'interface com.mycompany.mylocalbusiness locale, le JNDI EJB 3.0 présélectionné est :

  ```
  ejblocal:com.mycompany.mylocalbusiness
  ```

Pour l'interface com.mycompany.myremotebusiness éloignée, le JNDI EJB 3.0 présélectionné correspond à l'interface complète :

```
com.mycompany.myremotebusiness
```

Les liaisons d'application par défaut EJB 3.0 sont décrites dans l'emplacement suivant : Présentation des liaisons d'application EJB 3.0.

Integration Designer utilise le nom "abrégé" comme emplacement JNDI par défaut des EJB à l'aide du modèle de programmation de la version 3.0.

Remarque : Si la référence JNDI déployée de l'EJB cible est différente de l'emplacement de la liaison JNDI par défaut car un mappage personnalisé a été utilisé ou configuré, le nom JNDI cible doit être correctement spécifié. Vous pouvez spécifier le nom dans Integration Designer avant le déploiement ou, pour la liaison d'importation, vous pouvez modifier le nom dans la console d'administration (après le déploiement), pour qu'il corresponde au nom JNDI de l'EJB cible.

Pour plus d'informations sur la création de liaisons EJB, reportez-vous à la section relative à l'utilisation des liaisons EJB dans le Integration Designer centre de documentation.

Gestionnaire de données JAX-WS :

La liaison d'exportation Enterprise JavaBeans (EJB) utilise le gestionnaire de données JAX-WS pour transformer les objets métier de demande en paramètres d'objet Java et la valeur de retour d'objet Java en objet métier de réponse. La liaison d'exportation EJB utilise le gestionnaire de données JAX-WS pour transformer les EJB de demande en objets métier de demande et l'objet métier de réponse en valeur renvoyée.

Ce gestionnaire de données mappe les données de l'interface indiquée par SCA à l'interface Java de l'EJB cible (et vice versa) à l'aide de la spécification JAX-WS (Java API for XML Web Services) et de la spécification JAXB (Java Architecture for XML Binding).

Remarque : Le support actuel est limité aux spécifications JAX-WS 2.1.1 et JAXB 2.1.3.

Le gestionnaire de données spécifié au niveau de la liaison EJB est utilisé pour traiter les demandes, les réponses, les incidents et les exceptions d'exécution.

Remarque : Pour les erreurs, un gestionnaire de données propre peut être spécifié pour chaque erreur en indiquant la propriété de configuration faultBindingType. Cette dernière remplace la valeur spécifiée au niveau de la liaison EJB.

Le gestionnaire de données JAX-WS est utilisé par défaut lorsque la liaison EJB dispose d'une interface WSDL. Ce gestionnaire de données ne peut pas être utilisé pour transformer un message SOAP représentant un appel JAX-WS en objet de données.

La liaison d'importation EJB utilise un gestionnaire de données pour transformer un objet de données en matrice d'objets Java (Object[]). Lors des communications sortantes, le traitement suivant est effectué :

1. La liaison EJB définit le type et l'élément attendus ainsi que le nom de la méthode ciblée dans BindingContext, afin qu'ils soient identiques à ceux spécifiés dans le fichier WSDL.
2. La liaison EJB appelle la méthode de transformation pour l'objet de données nécessitant une transformation des données.
3. Le gestionnaire de données renvoie un tableau Object[] représentant les paramètres de la méthode (listés dans l'ordre suivant lequel ils sont définis dans la méthode).
4. La liaison EJB utilise le tableau Object[] pour appeler la méthode sur l'interface EJB cible.

La liaison prépare également un tableau Object[] pour traiter la réponse à partir de l'appel EJB.

- Le premier élément dans l'objet[] correspond à la valeur renvoyée par l'appel de méthode Java.

- Les valeurs suivantes représentent les paramètres d'entrée de la méthode.

Ces dernières sont nécessaires pour prendre en charge les paramètres de type entrée/sortie et sortie.

Pour les paramètres de type sortie, les valeurs doivent être renvoyées dans l'objet de données de réponse.

Le gestionnaire de données traite et transforme les valeurs rencontrées dans l'objet[], puis renvoie une réponse à l'objet de données.

Le gestionnaire de données prend en charge xs:AnyType, xs:AnySimpleType, and xs:Any, ainsi que d'autres types de données XSD. Pour activer la prise en charge de xs:Any, utilisez **@XmlAnyElement (lax=true)** pour la propriété JavaBeans dans le code Java, comme illustré dans l'exemple suivant :

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Cela permet de transformer l'objet de propriété dans TestType en champ xs:any. La valeur de classe Java utilisée dans le champ xs:any doit inclure l'annotation **@XmlAnyElement**. Par exemple, si Adresse est la classe Java utilisée pour remplir la matrice d'objets, elle doit comporter l'annotation **@XmlRootElement**.

Remarque : Pour personnaliser le mappage du type XSD aux types Java définis par la spécification JAX-WS, modifiez les annotations JAXB selon les besoins de votre entreprise. Le gestionnaire de données JAX-WS prend en charge xs:any, xs:anyType et xs:anySimpleType.

Les restrictions suivantes s'appliquent au gestionnaire de données JAX-WS :

- Le gestionnaire de données ne prend pas en charge l'annotation **@WebParam** de l'attribut d'en-tête.
- L'espace de noms pour les fichiers de schéma des objet métiers (fichiers XSD) n'inclut pas le mappage par défaut du nom de package Java. L'annotation **@XMLSchema** dans package-info.java ne fonctionne pas non plus. La seule manière de créer un fichier XSD avec un espace de nom consiste à utiliser les annotations **@XmlType** et **@XmlRootElement**. **@XmlRootElement** définit l'espace de nom cible pour l'élément global dans les types de JavaBeans.
- L'assistant d'importation EJB ne crée pas de fichiers XSD pour les classes non associées. La version 2.0 ne prend pas en charge l'annotation **@XmlSeeAlso**. Par conséquent, si la classe enfant n'est pas directement référencée par la classe parente, aucun fichier XSD n'est créé. La solution à ce problème consiste à exécuter SchemaGen pour ces classes enfants.

  SchemaGen est un utilitaire de ligne de commande (situé dans le répertoire *Rép_base_installation_WPS*/bin) fourni pour créer des fichiers XSD pour un bean donné. Ces fichiers XSD doivent être manuellement copiés vers le module pour que la solution fonctionne.

Sélecteur d'erreur EJB :

Le sélecteur d'erreur EJB détermine si un appel EJB a engendré une erreur, une exception d'exécution ou une réponse correcte.

Si une erreur est détectée, le sélecteur d'erreurs EJB renvoie le nom de l'erreur native à l'environnement d'exécution de la liaison afin que le gestionnaire de données JAX-WS puisse convertir l'objet exception en objet métier d'erreur.

En cas de réponse positive (pas d'erreur), la liaison d'importation EJB assemble une matrice d'objets Java (Object[]) pour renvoyer les valeurs.

- Le premier élément dans l'objet[] correspond à la valeur renvoyée par l'appel de méthode Java.
- Les valeurs suivantes représentent les paramètres d'entrée de la méthode.

Ces dernières sont nécessaires pour prendre en charge les paramètres de type entrée/sortie et sortie.

Pour les scénarios d'exception, la liaison assemble un objet[] et le premier élément représente l'exception générée par la méthode.

Le sélecteur d'erreurs peut renvoyer l'une des valeurs suivantes :

Tableau 35. Valeurs renvoyées

Type	Valeur renvoyée	Description
Erreur	**ResponseType.FAULT**	Renvoyée si la matrice Object[] transmise contient un objet d'exception.
Exception d'exécution	**ResponseType.RUNTIME**	Renvoyée si l'objet d'exception ne correspond à aucun des types d'exception déclarés sur la méthode.
Réponse normale	**ResponseType.RESPONSE**	Renvoyée dans tous les autres cas.

Si le sélecteur d'erreurs renvoie la valeur **ResponseType.FAULT**, le nom d'erreur natif est renvoyé. Ce nom est utilisé par la liaison pour déterminer le nom d'erreur WSDL correspondant à partir du modèle et appeler le gestionnaire de données d'erreur approprié.

Sélecteur de fonction EJB :

Les liaisons EJB utilisent un sélecteur de fonction d'importation (pour le traitement sortant) un sélecteur de fonction d'exportation (pour le traitement entrant) afin de déterminer la méthode EJB à appeler.

Sélecteur de fonction d'importation

Pour le traitement sortant, le sélecteur de fonction d'importation détermine le type de méthode EJB en fonction du nom de l'opération appelée par le composant SCA connecté à l'importation EJB. Le sélecteur de fonction recherche l'annotation @WebMethod sur la classe Java annotée par JAS-WS et générée par Integration Designer pour déterminer le nom de l'opération cible associée.

- Si l'annotation @WebMethod est présente, le sélecteur de fonction utilise l'annotation @WebMethod pour déterminer le mappage correct de la méthode Java pour la méthode WSDL.
- Si l'annotation @WebMethod est manquante, le sélecteur de fonction suppose que le nom de la méthode Java est identique au nom de l'opération appelée.

Remarque : Ce sélecteur de fonction n'est valide que pour une interface de type WSDL sur une importation EJB et non pour une interface de type Java sur une importation EJB.

Le sélecteur de fonction renvoie un objet java.lang.reflect.Method qui représente la méthode de l'interface EJB.

Le sélecteur de fonction utilise une matrice d'objets Java (Object[]) pour conserver la réponse de la méthode cible. Le premier élément de la matrice Object[] est une méthode Java possédant le nom du WSDL et le second correspond à l'objet métier en entrée.

Sélecteur de fonction d'exportation

Pour le traitement entrant, le sélecteur de fonction d'exportation détermine la méthode cible à appeler à partir de la méthode Java.

Le sélecteur de fonction d'exportation mappe le nom de l'opération Java appelée par le client EJB au nom de l'opération dans l'interface du composant cible. Le nom de la méthode est renvoyé sous forme de chaîne et résolu par l'environnement d'exécution de SCA en fonction du type d'interface du composant cible.

Liaisons EIS :

Les liaisons de système d'information d'entreprise (EIS) permettent la connectivité entre les composants SCA et un système EIS externe. Cette communication est accomplie à l'aide des exportations et des importations EIS qui prennent en charge les adaptateurs de ressources JCA 1.5 et les adaptateurs Websphere.

Vos composants SCA peuvent exiger le transfert de données vers ou à partir d'un EIS externe. Lorsque vous créez un module SCA exigeant une telle connectivité, vous incluez (outre le composant SCA) une importation ou exportation avec une liaison EIS pour communiquer avec un EIS externe spécifique.

Les adaptateurs de ressources dans IBM Integration Developer sont utilisés dans le contexte d'une importation ou d'une exportation. Vous développez une importation ou une exportation à l'aide de l'assistant de service externe puis, lors du développement vous incluez l'adaptateur de ressources. Une importation EIS permettant à votre application d'appeler un service sur un système EIS ou une exportation EIS permettant à une application sur un système EIS d'appeler un service développé dans IBM Integration Developer sont créées avec un adaptateur de ressources. Par exemple, vous pourriez créer une importation avec un adaptateur JD Edwards pour appeler un service sur le système JD Edwards.

Lorsque vous utilisez un assistant de service externe, les informations de liaison EIS sont créées pour vous. Vous pouvez également utiliser un autre outil, l'éditeur d'assemblage, pour ajouter ou modifier des informations de liaison. Pour plus d'informations, voir Accès aux services externes à l'aide des adaptateurs.

Une fois que le module SCA contenant la liaison EIS est déployé sur le serveur, vous pouvez utiliser la console d'administration pour afficher les informations relatives à la liaison ou pour la configurer.

Présentation des liaisons EIS :

Lorsqu'elle est utilisée avec un adaptateur de ressources JCA, la liaison EIS (Enterprise Information System) permet d'accéder à des services sur un système d'information d'entreprise ou de rendre vos services disponibles auprès de l'EIS.

L'exemple suivant illustre la manière dont un module SCA appelé ContactSyncModule synchronise les informations de contact entre un système Siebel et un système SAP.

1. Le composant SCA appelé ContactSync écoute (via une exportation d'application EIS appelée Siebel Contact) les modifications apportées aux contacts Siebel.
2. Le composant ContactSync lui-même utilise une application SAP (via une importation d'application EIS) afin de mettre à jour les informations de contact SAP en conséquence.

Comme les structures de données utilisées pour le stockage de contacts sont différentes dans les systèmes Siebel et SAP, le composant SCA ContactSync doit assurer le mappage.

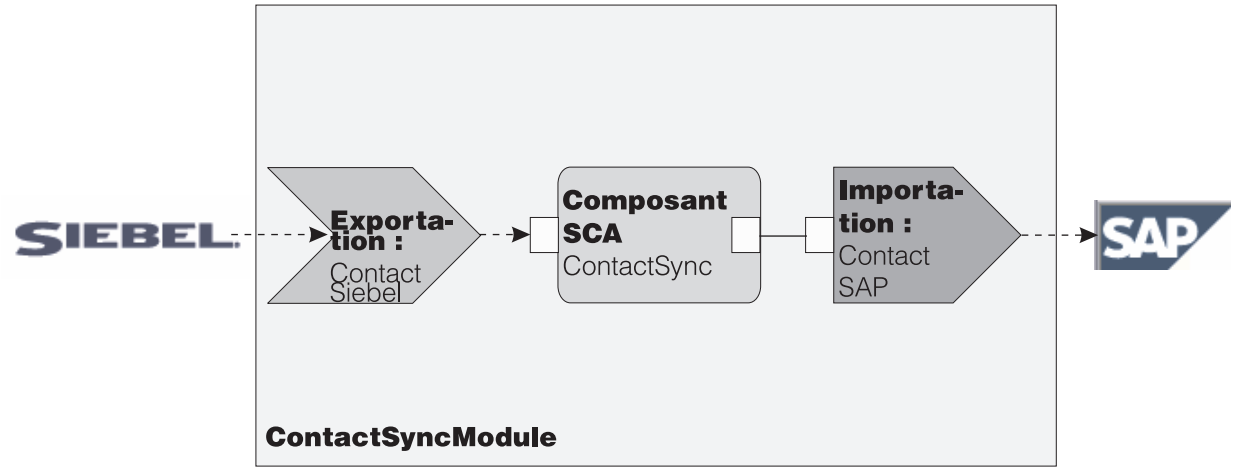


Figure 31. Flux provenant d'un système Siebel vers un système SAP

L'exportation Siebel Contact et l'importation SAP Contact ont les adaptateurs de ressources appropriés configurés.

Principales fonctionnalités des liaisons EIS :

Une importation EIS est une importation SCA (Service Component Architecture) qui permet aux composants du module SCA d'utiliser les applications EIS définies hors du module SCA. Une importation EIS permet de transférer des données d'un composant SCA vers un système EIS externe. Une exportation EIS permet de transférer des données d'un système EIS externe vers un module SCA.

Importations

La fonction de l'importation EIS est de combler l'écart entre les composants SCA et les systèmes EIS externes. Les applications externes peuvent être traitées comme des importations EIS. Dans ce cas, l'importation EIS envoie des données au système EIS externe et reçoit éventuellement des données en réponse.

L'importation EIS fournit aux composants SCA une vue uniforme des applications externes au module. Ceci permet aux composants de communiquer avec un système EIS externe, tel que SAP, Siebel ou PeopleSoft, via un modèle de programmation SCA homogène.

Côté client de l'importation, une interface est affichée par l'application d'importation EIS, laquelle comporte une ou plusieurs méthodes, chacune prenant des objets de données servant d'arguments et de valeurs de retour. Côté implémentation, une interface client commune (Common Client Interface - CCI) est implémentée par un adaptateur de ressources.

L'implémentation d'exécution d'une importation EIS connecte l'interface côté client et l'interface CCI. L'importation mappe l'appel de la méthode de l'interface à l'appel sur l'interface CCI.

Des liaisons sont créées à trois niveaux : une liaison d'interface, qui utilise les liaisons de méthode incluses, lesquelles utilisent les liaisons de données.

La liaison d'interface associe l'interface de l'importation à la connexion au système EIS qui fournit l'application. Ceci reflète le fait que l'ensemble d'applications, représenté par l'interface, est fourni par l'instance spécifique du système EIS et que la connexion permet l'accès à cette instance. L'élément de liaison contient des propriétés avec suffisamment d'informations pour créer la connexion (ces propriétés font partie de l'instance javax.resource.spi.ManagedConnectionFactory).

La liaison de méthode associe la méthode à l'interaction spécifique avec le système EIS. Pour JCA, cette interaction se caractérise par l'ensemble de propriétés de l'implémentation de l'interface javax.resource.cci.InteractionSpec. L'élément d'interaction de la liaison de méthode contient ces propriétés, ainsi que le nom de la classe, fournissant ainsi assez d'informations pour permettre l'interaction. La liaison de méthode utilise des liaisons de données décrivant le mappage de l'argument et du résultat de la méthode d'interface à la représentation EIS.

Le scénario d'exécution d'une importation EIS est le suivant :

1. La méthode de l'interface d'importation est appelée à l'aide du modèle de programmation SCA.
2. La demande, qui parvient au gestionnaire d'importation EIS, contient le nom de la méthode et ses arguments.
3. Tout d'abord, l'importation crée une implémentation de liaison d'interface. Ensuite, à partir des données de la liaisond'importation, il crée une ConnectionFactory et associe les deux. Autrement dit, l'importation appellesetConnectionFactory sur la liaison d'interface.
4. L'implémentation de liaison de méthode correspondant à la méthode appelée est créée.
5. L'instance javax.resource.cci.InteractionSpec est créée et remplie, puis les liaisons de données sont utilisées pour lier les arguments de la méthode à un format lisible par l'adaptateur de ressource.
6. L'interface CCI permet d'appliquer l'interaction.
7. Lors du retour de l'appel, la liaison de données permet de créer le résultat de l'appel et de renvoyer ce résultat au demandeur.

Exportations

La fonction de l'exportation EIS est de combler l'écart entre un composant SCA et un système EIS externe. Les applications externes peuvent être traitées comme des exportations EIS. Dans ce cas, l'application externe envoie ses données sous la forme de notifications périodiques. Une exportation EIS peut être considérée comme une application d'abonnement qui écoute une demande externe provenant d'un EIS. Le composant SCA qui utilise l'exportation EIS la voit comme une application locale.

L'exportation EIS fournit aux composants SCA une vue uniforme des applications externes au module. Ceci permet aux composants de communiquer avec un système EIS, tel que SAP, Siebel ou PeopleSoft, via un modèle de programmation SCA homogène.

L'exportation comporte une implémentation d'écouteur recevant des demandes du système EIS. L'écouteur implémente une interface d'écouteur spécifique à l'adaptateur de ressources. L'exportation contient également une interface d'implémentation de composant, exposée au système EIS via l'exportation.

L'implémentation d'exécution d'une exportation EIS connecte l'écouteur et l'interface d'implémentation de composant. L'exportation mappe la demande EIS à l'appel de l'opération appropriée sur le composant. Des liaisons sont créées à trois niveaux : une liaison d'écouteur, qui utilise une liaison de méthode native incluse, laquelle utilise une liaison de données.

La liaison d'écouteur associe l'écouteur qui reçoit des demandes au composant affiché via l'exportation. La définition d'exportation contient le nom du composant ; l'environnement d'exécution le localise et lui transmet des demandes.

La liaison de méthode native associe la méthode native ou le type d'événement reçu par l'écouteur à l'opération implémentée par le composant affiché au moyen de l'exportation. Il n'y a aucune relation entre la méthode appelée sur l'écouteur et le type d'événement ; tous les événements arrivent par l'intermédiaire d'une ou de plusieurs méthodes de l'écouteur. La liaison de méthode native utilise le sélecteur de fonction défini dans l'exportation pour extraire le nom de la méthode native des données entrantes et des liaisons de données pour lier le format de données du système EIS à un format lisible par le composant.

Le scénario d'exécution d'une exportation EIS est le suivant :

1. La demande EIS déclenche l'appel de la méthode sur l'implémentation de l'écouteur.
2. L'écouteur localise et appelle l'application d'exportation en lui transmettant tous les arguments d'appel.
3. L'exportation crée l'implémentation de liaison d'écouteur.
4. L'exportation instancie le sélecteur de fonction et le définit sur la liaison d'écouteur.
5. L'exportation initialise les liaisons de méthode native et les ajoute à la liaison de l'écouteur. Pour chaque liaison de méthode native, les liaisons de données sont également initialisées.
6. L'exportation appelle la liaison de l'écouteur.
7. Cette liaison localise les composants exportés et utilise le sélecteur de fonction pour extraire le nom de la méthode native.
8. Ce nom est utilisé pour localiser la liaison de méthode native, qui peut alors appeler le composant cible.

Le type d'interaction de l'adaptateur permet à la liaison d'exportation EIS d'appeler le composant cible de manière asynchrone (par défaut) ou synchrone.

Adaptateurs de ressources

Vous développez une importation ou une exportation à l'aide de l'assistant de service externe puis, lors du développement, vous incluez un adaptateur de ressources. Les adaptateurs livrés avec IBM Integration Designer pour accéder aux systèmes CICS, IMS, JD Edwards, PeopleSoft, SAP et Siebel sont réservés à des fins de développement et de test uniquement. Autrement dit, vous les utilisez pour développer et tester vos applications.

Après le déploiement de votre application, vous avez besoin d'adaptateurs d'exécution sous licence pour exécuter votre application. Toutefois, lorsque vous créez votre service, vous pouvez y intégrer l'adaptateur. La licence pour votre adaptateur peut vous permettre d'utiliser l'adaptateur intégré comme adaptateur d'exécution sous licence. Ces adaptateurs sont compatibles avec la norme JCA 1.5 (Java EE Connector Architecture). JCA, standard ouvert, est la norme Java EE pour la connectivité EIS. JCA fournit un cadre géré ; la qualité de service (QoS) est assurée par le serveur d'applications qui offre aux transactions la sécurité et la gestion du cycle de vie. Ces adaptateurs sont également compatibles avec la spécification Enterprise Metadata Discovery, à l'exception de l'adaptateur de ressources IBM CICS ECI et du connecteur IBM IMS pour Java.

Les adaptateurs WebSphere Business Integration Adapters, un ancien jeu d'adaptateurs, sont également pris en charge par l'assistant.

Ressources Java EE

Le module EIS, un module SCA qui repose sur le pattern des modules EIS, peut être déployé sur la plateforme Java EE.

Le déploiement d'un module EIS sur la plateforme Java EE génère une application prête à être exécutée, encapsulée dans un fichier EAR et déployée sur le serveur. Tous les artefacts et ressources Java EE sont créés ; l'application est configurée et prête pour l'exécution.

Propriétés dynamiques des spécifications d'interaction et de connexion JCA :

La liaison EIS peut accepter des données en entrée pour lesspécifications InteractionSpec et ConnectionSpec spécifiées, en utilisant un objet de données enfant bien défini qui accompagne la charge. Ceci permet des interactions demande-réponse dynamiques avec un adaptateur de ressources par le biais de InteractionSpec et l'authentification des composants par le biais de ConnectionSpec.

L'interface javax.cci.InteractionSpec transmet des informations sur le mode de traitement de la demande d'interaction avec l'adaptateur de ressources. Elle comporte également des informations sur l'accomplissement de l'interaction après la demande. Ces communications bidirectionnelles par le biais des interactions sont parfois appelées *conversations*.

La liaison EIS s'attend à ce que la charge qui sera l'argument de l'adaptateur de ressources contienne un objet de données enfant appelé **properties**. Cette objet de données de propriétés contient des paires nom/valeur, avec le nom des propriétés de la spécification d'interaction dans un format particulier. Les règles de formatage sont les suivantes :

- Les noms doivent commencer par le préfixe **IS**, suivi du nom de propriété. Par exemple, une spécification d'interaction avec une propriété JavaBeans appelée **InteractionId** doit spécifier le nom de la propriété sous la forme **ISInteractionId**.
- La paire nom/valeur représente le nom et la valeur du type simple de propriété de la spécification d'interaction.

Dans cet exemple, une interface spécifie que l'entrée d'une opération est un objet de données **Compte**. Cette interface appelle une application de liaison d'importation EIS dans le but d'envoyer et de recevoir une propriété InteractionSpec dynamique appelée **workingSet** avec la valeur **xyz**.

Le graphique métier ou les objets métier du serveur contiennent un objet métier **properties** sous-jacent qui permet l'envoi de données propres au protocole avec la charge. Cet objet métier**properties** est intégré, ce qui fait qu'il n'est pas nécessaire de le spécifier dans le schéma XML lors de la construction d'un objet métier. Il convient seulement de le créer et de l'utiliser. Si vous avez défini vos propres types de données sur la base du schéma XML, vous devez spécifier un élément **properties** qui contient les paires nom/valeur attendues.

```
BOFactory dataFactory = (BOFactory) \
serviceManager.locateService("com/ibm/websphere/bo/BOFactory");
 //Wrapper for doc-lit wrapped style interfaces,
 //skip to payload for non doc-lit
 DataObject docLitWrapper = dataFactory.createByElement /
 ("http://mytest/eis/Account", "AccountWrapper");
```

Créez la charge.

```
DataObject account = docLitWrapper.createDataObject(0);
  DataObject accountInfo = account.createDataObject("AccountInfo");
  //Perform your setting up of payload


  //Construct properties data for dynamic interaction

  DataObject properties = account.createDataObject("properties");
```

Pour le nom workingSet, définissez la valeur attendue (**xyz**).

```
properties.setString("ISworkingSet", "xyz");


  //Invoke the service with argument

  Service accountImport = (Service) \
  serviceManager.locateService("AccountOutbound");
  DataObject result = accountImport.invoke("createAccount", docLitWrapper);

  //Get returned property
  DataObject retProperties = result.getDataObject("properties");

  String workingset = retProperties.getString("ISworkingSet");
```

Vous pouvez utiliser des propriétés ConnectionSpec pour l'authentification des composants dynamiques. Les mêmes règles s'appliquent que ci-dessus, sauf que le préfixe du nom de propriété doit être **CS** au lieu de **IS**. Les propriétés ConnectionSpec ne sont pas bidirectionnelles. Le même objet de données **properties** peut contenir à la fois des propriétés IS et CS.

Pour utiliser les propriétés ConnectionSpec, définissez **resAuth** spécifié dans la liaison d'importation sur **Application**. Assurez-vous également que l'adaptateur de ressources prend en charge l'autorisation de composant. Pour plus de détails, reportez-vous au chapitre 8 du document J2EE Connector Architecture Specification.

Clients externes et liaisons EIS :

Le serveur peut envoyer des messages à, ou recevoir des messages de clients externes par le biais de liaisons EIS.

Un client externe, par exemple un portail Web ou un système EIS, doit envoyer un message à un module SCA dans le serveur ou doit être appelé par un composant à partir du serveur.

Le client appelle l'importation EIS comme avec toute autre application, à l'aide soit de l'interface DII (Dynamic Invocation Interface), soit de l'interface Java.

1. Le client externe crée une instance de l'interface ServiceManager et recherche l'importation EIS à l'aide de son nom de référence. Le résultat de la recherche est une implémentation de l'interface de service.
2. Le client crée un argument d'entrée, un objet de données générique, créé dynamiquement à l'aide du schéma de l'objet de données. Cette étape est effectuée à l'aide de l'implémentation de l'interface Service Data Object DataFactory.
3. Le client externe appelle l'EIS et obtient les résultats requis.

Le client a également la possibilité d'appeler l'importation EIS par le biais de l'interface Java.

1. Le client externe crée une instance de l'interface ServiceManager et recherche l'importation EIS à l'aide de son nom de référence. Le résultat de la recherche est une interface Java de l'importation EIS.
2. Le client crée un argument d'entrée et un objet de données typé.
3. Le client appelle l'EIS et obtient les résultats requis.

L'interface d'exportation EIS définit l'interface du composant SCA exporté qui est accessible aux applications EIS externes. Cette interface peut être considérée comme l'interface qu'une application externe (comme SAP ou PeopleSoft) va appeler par le biais de l'implémentation de l'environnement d'exécution de l'application d'exportation EIS.

L'exportation utilise EISExportBinding pour lier les services exportés à l'application EIS externe. Elle vous permet d'abonner une application contenue dans votre module SCA pour écouter les demandes de service EIS. La liaison d'exportation EIS spécifie le mappage entre la définition des événements entrants telle qu'elle est comprise par l'adaptateur de ressources (à l'aide des interfaces Java EE Connector Architecture) et l'appel des opérations SCA.

L'interface EISExportBinding exige que les services EIS externes soient basés sur des contrats entrants de Java EE Connector Architecture 1.5. L'interface EISExportBinding exige qu'un gestionnaire de données ou une liaison de données soit spécifié(e) au niveau de la liaison ou au niveau de la méthode.

Liaisons JMS :

Un fournisseur JMS (Java Message Service) active la messagerie en fonction du modèle de programmation et de l'API JMS (Java Messaging Service). Il fournit des fabriques de connexions JMS pour créer des connexions pour des destinations JMS et pour envoyer et recevoir des messages.

Les liaisons JMS peut être utilisé lorsque vous interagissez avec la liaison du fournisseur du bus d'intégration de services (SIB) et si elles sont compatibles avec JMS et JCA 1.5.

Les liaisons d'importation et d'exportation JMS permettent à un module SCA (Service Component Architecture) d'appeler des systèmes JMS externes et d'en recevoir des messages.

Les liaisons d'importation et d'exportation JMS permettent une intégration aux applications JMS en utilisant le fournisseur JMS SIB basé sur JCA 1.5 qui fait partie de WebSphere Application Server. Les autres adaptateurs de ressources JMS basés sur JCA 1.5 ne sont pas pris en charge

Présentation des liaisons JMS :

Les liaisons JMS assurent la connectivité entre l'environnement SCA (Service Component Architecture) et les systèmes JMS.

Liaisons JMS

Les principaux composants des liaisons d'importation et d'exportation JMS sont les suivants :

- Adaptateur de ressources : assure une connectivité bidirectionnelle gérée entre un module SCA et des systèmes JMS externes
- Connexions : encapsulent une connexion virtuelle entre un client et une application fournisseur
- Destinations : utilisées par un client pour spécifier la cible des messages produits ou la source des messages utilisés
- Données d'authentification : permettent de sécuriser l'accès à la liaison

Principales fonctionnalités des liaisons JMS

En-têtes spéciaux

Des propriétés d'en-tête spéciales sont utilisées dans les importations et les exportations JMS pour indiquer à la cible comment traiter le message.

Par exemple, TargetFunctionName mappe la méthode native à la méthode d'opération.

Ressources Java EE

Plusieurs ressources Java EE sont créées lorsque les importations et les exportations JMS sont déployées dans un environnement Java EE.

`ConnectionFactory`
Utilisée par les clients pour créer une connexion au fournisseur JMS.

`ActivationSpec`
Utilisé par les importations pour la réception de la réponse à une demande et par les exportations pour la configuration endpoint des noeuds finaux de message qui représentent les écouteurs de messages dans leurs interactions avec le système de messagerie.

`Destinations`

- Destination d'envoi : Destination à laquelle est envoyé le message de demande ou sortant (importation) ou le message de réponse (exportation), si la valeur n'est pas remplacée par la zone d'en-tête JMSReplyTo du message entrant.
- Destination de réception : emplacement où doit être placé le message entrant ; dans les importations, il s'agit d'une réponse, dans les exportations, il s'agit d'une demande.
- Destination de rappel : Destination du système JMS SCA utilisée pour stocker les informations de corrélation. Vous ne devez pas procéder à des opérations de lecture ou d'écriture sur cette destination.

La tâche d'installation crée ConnectionFactory et trois destinations. En outre, elle crée ActivationSpec pour permettre à l'écouteur de messages d'exécution d'écouter les réponses sur la destination de réception. Les propriétés de ces ressources sont définies dans le fichier d'importation ou d'exportation.

Intégration JMS et adaptateurs de ressources :

Le service JMS (Java Message Service) assure une intégration par le biais d'un adaptateur de ressources JMS disponible basé sur JCA 1.5. La prise en charge complète de l'intégration JMS est assurée pour l'adaptateur de ressources JMS du bus d'intégration de services (SIB).

Utilisez un fournisseur JMS pour JCA 1.5 lorsque vous souhaitez une intégration avec un système JMS externe conforme au JCA 1.5. Les services externes conformes au JCA 1.5 peuvent recevoir et envoyer des messages à intégrer avec vos composants SCA à l'aide de l'adaptateur de ressources JMS SIB.

L'emploi d'adaptateurs de ressources JCA 1.5 propres à un fournisseur n'est pas pris en charge.

Liaisons d'importation et d'exportation JMS :

Vous pouvez faire interagir les modules SCA avec les services fournis par des applications JMS externes à l'aide des liaisons d'importation et d'exportation JMS.

Liaisons d'importation JMS

Les connexions avec le fournisseur JMS associé de destinations JMS sont créées à l'aide d'une fabrique de connexions JMS. Utilisez les objets d'administration de fabrique de connexions afin de gérer des fabriques de connexion JMS pour le fournisseur de messagerie par défaut.

L'interaction avec les systèmes JMS externes comprend l'utilisation des destinations pour l'envoi des requêtes et la réception des réponses.

Deux types de scénarios d'utilisation pour l'importation JMS sont pris en charge en fonction du type d'opération appelée :

- Unidirectionnel : l'importation JMS place un message sur la destination d'envoi configurée dans la liaison d'importation. Rien n'est défini dans la zone replyTo de l'en-tête JMS.
- Bidirectionnel (demande-réponse) : l'importation JMS place un message sur la destination d'envoi et conserve ensuite la réponse reçue depuis le composant SCA.

  La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut), ou à partir de l'ID de corrélation de message de demande. La liaison d'importation peut également être configurée pour utiliser une destination de réponse dynamique temporaire afin d'établir une corrélation entre les réponses et les demandes. Une destination temporaire est créée pour chaque demande et l'importation l'utilise pour recevoir la réponse.

  La destination receive est définie dans la propriété d'en-tête replyTo du message sortant. Un écouteur de messages est déployé pour écouter sur la destination de réception et, dès qu'une réponse est reçue, l'écouteur de messages transmet la réponse au composant.

Pour les scénarios d'utilisation unidirectionnel et bidirectionnel, les propriétés d'en-tête dynamique et statique peuvent être spécifiées. Les propriétés statiques peuvent être définies à partir de la liaison de méthode d'importation JMS. Certaines de ces propriétés revêtent des significations particulières pour l'environnement d'exécution JMS SCA.

Il est important de noter que JMS est une liaison asynchrone. Si un composant appelant appelle une importation JMS de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service JMS.

La figure 32 montre comment l'importation est liée au service externe.

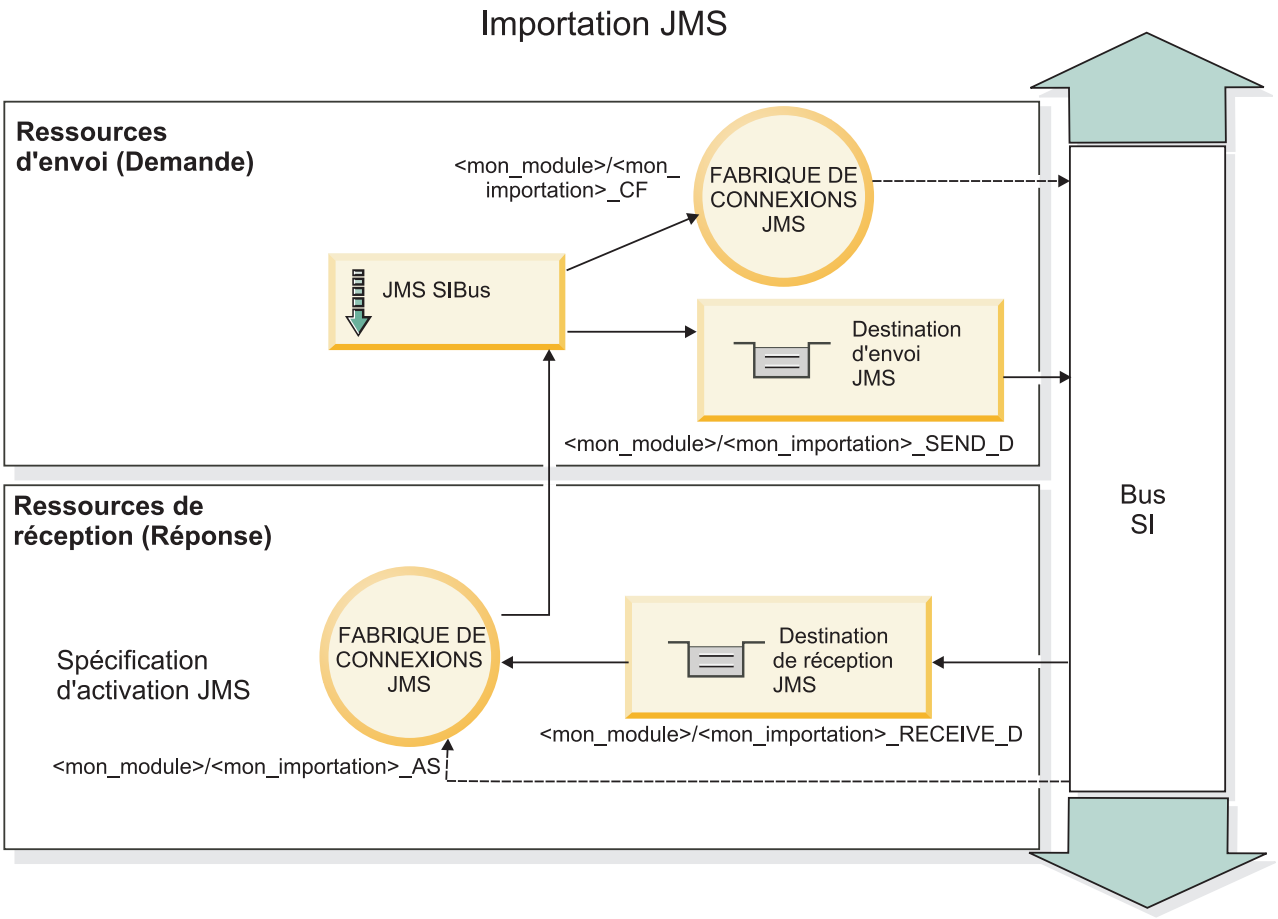


Figure 32. Ressources de liaisons d'importation JMS

Liaisons d'exportation JMS

Les liaisons d'exportation JMS offrent les moyens aux modules SCA de fournir des services aux applications JMS externes.

La connexion qui fait partie d'une exportation JMS est une spécification d'activation.

Une exportation JMS comporte des destinations d'envoi et de réception.

- La destination receive est le lieu de réception du message entrant destiné au composant cible.
- La destination send est celle à laquelle la réponse sera envoyée, sauf si le message entrant l'a remplacé en utilisant la propriété d'en-tête replyTo.

Un écouteur de messages est déployé pour écouter les demandes parvenant à la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse. La destination spécifiée dans la zone replyTo du message entrant remplace la destination spécifiée dans send.

La figure 33 illustre la manière dont le demandeur externe est lié à l'exportation.

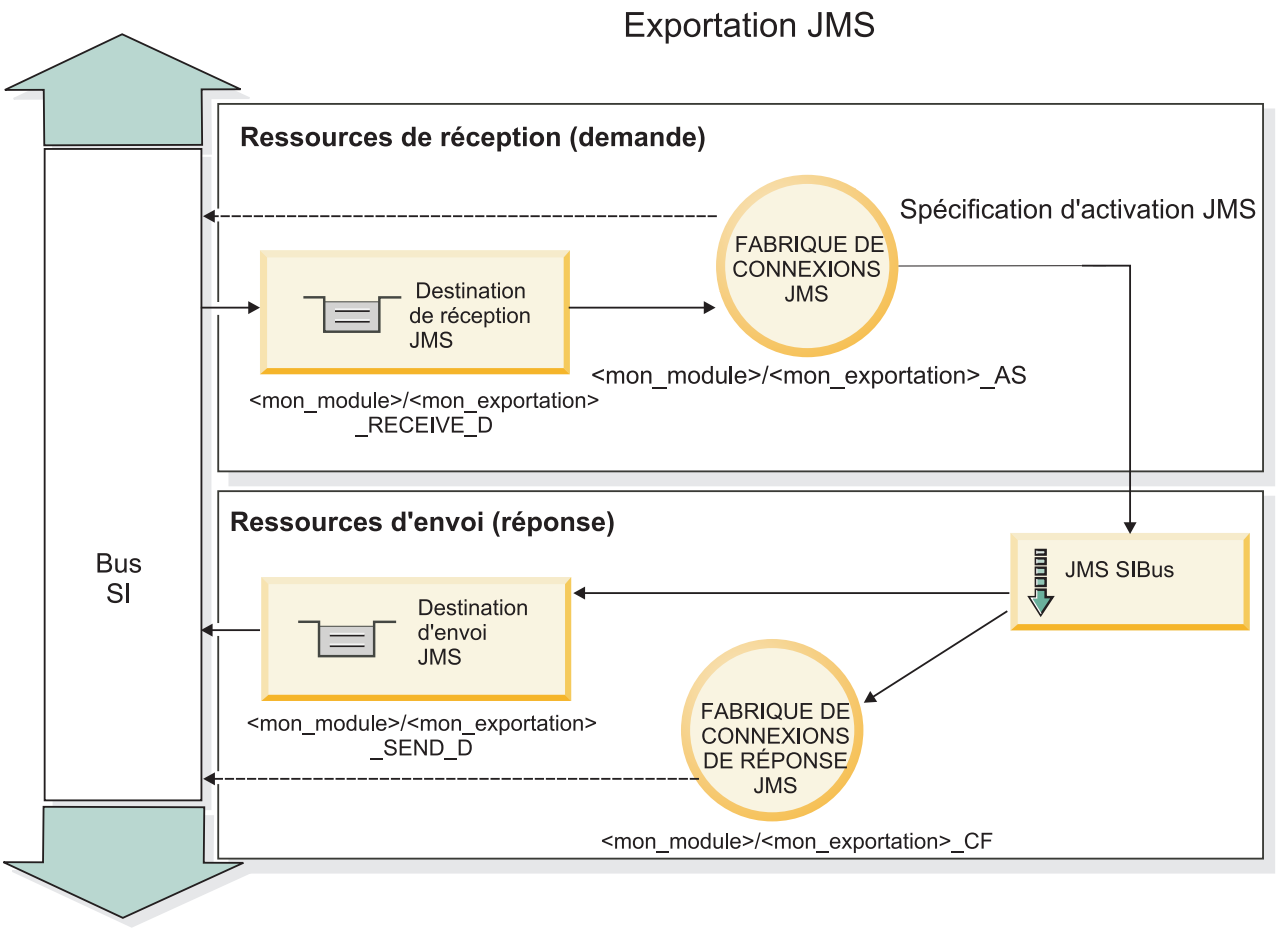


Figure 33. Ressources de liaisons d'exportation JMS

En-têtes JMS :

Un message JMS contient deux types d'en-têtes : l'en-tête de système JMS et plusieurs propriétés JMS. Il est possible d'accéder aux deux types d'en-tête depuis un module de médiation dans l'objet SMO (Service Message Object) ou en utilisant l'API ContextService.

En-tête système JMS

L'en-tête système JMS est représenté dans l'objet SMO par l'élément JMSHeader qui contient toutes les zones généralement présentes dans un en-tête JMS. Bien qu'elles puissent être modifiées dans la médiation (ou ContextService), certaines zones d'en-tête système JMS définies dans l'objet SMO ne seront pas propagées dans le message JMS sortant puisqu'elles sont remplacées par des valeurs statiques ou système.

Les zones clés de l'en-tête système JMS qui peuvent être mises à jour dans une médiation (ou ContextService) sont :

- **JMSType** et **JMSCorrelationID** : valeurs des propriétés de l'en-tête du message prédéfinies spécifiques
- **JMSDeliveryMode** : valeurs du mode de livraison (persistant (par défaut) ou non persistant)
- **JMSPriority** : valeur de la priorité (0 à 9 ; la valeur par défaut est JMS_Default_Priority)

Propriétés JMS

Les propriétés JMS sont représentées dans l'objet SMO en tant qu'entrées dans la liste Propriétés. Les propriétés peuvent être ajoutées, mises à jour ou supprimées dans une médiation ou en utilisant l'API ContextService.

Elles peuvent également être définies de manière statique dans la liaison JMS. Les propriétés définies de manière statique remplacent les paramètres (portant le même nom) qui sont définis de manière dynamique.

Les propriétés utilisateur propagées à partir d'autres liaisons (par exemple, une liaison HTTP) correspondront à une sortie dans la liaison JMS comme les propriétés JMS.

Paramètres de propagation d'en-tête

La propagation des propriétés et des en-têtes système JMS depuis le message JMS entrant vers les composants en aval ou depuis les composants en amont vers le message JMS sortant peut être contrôlée par l'indicateur Propagate Protocol Header.

Lorsque Propagate Protocol Header est défini, les informations d'en-tête peuvent circuler vers le message ou vers le composant cible, comme indiqué dans la liste suivante :

- Requête d'exportation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.
- Réponse d'exportation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS.
- Requête d'importation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS.
- Réponse d'importation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.

Schéma de corrélation des destinations de réponse dynamique temporaires JMS :

Le schéma de corrélation des destinations de réponse dynamique temporaires crée une rubrique ou une file d'attente dynamique unique pour chaque demande envoyée.

La destination de réponse statique spécifiée dans l'importation permet de déterminer la nature de la rubrique ou de la file d'attente de destination temporaire. Elle est définie dans la zone **ReplyTo** de la demande et l'importation JMS écoute les réponses sur cette destination. Lors de la réception d'une réponse, cette dernière est replacée dans la file d'attente de la destination des réponses statiques pour un traitement asynchrone. La zone **CorrelationID** de la réponse n'est pas utilisée et n'a pas besoin d'être définie.

Incidents transactionnels

Si une destination dynamique temporaire est utilisée, la réponse doit être consommée dans la même unité d'exécution que la réponse envoyée. La demande doit être envoyée en dehors de la transaction globale et doit être validée avant d'être reçue par le service dorsal et avant qu'une réponse ne soit renvoyée.

Persistance

Les files d'attente dynamiques temporaires sont des entités dont la durée de vie est courte et ne garantissent pas le même niveau de persistance que celui associé à une rubrique ou une file d'attente statique. Une rubrique ou une file d'attente dynamique temporaire ne survit pas à un redémarrage du serveur, tout comme les messages. Une fois que le message a été replacé en file d'attente dans la destination de réponse statique, elle conserve la persistance définie dans le message.

Délai d'attente

L'importation attend de recevoir la réponse sur la destination de réponse dynamique temporaire pendant un certain temps. Ce délai est extrait du qualifiant de délai d'expiration des réponses SCA, s'il est défini. Dans le cas contraire, le délai par défaut est de 60 secondes. En cas de dépassement de ce délai, l'importation génère une erreur `ServiceTimeoutRuntimeException`.

Clients externes :

Le serveur peut envoyer des messages à, ou recevoir des messages de clients externes par le biais de liaisons JMS.

Un client externe (comme un portail Web ou un système d'information d'entreprise) peut envoyer un message à un module SCA dans le serveur ou peut être appelé par un composant à partir du serveur.

Les composants d'exportation JMS déploient des écouteurs des messages pour écouter les demandes entrantes sur la destination receive spécifié dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si l'application appelée fournit une réponse. Ainsi, un client externe est en mesure d'appeler des applications avec la liaison d'exportation.

Les importations JMS interagissent avec les clients externes en envoyant des messages aux files d'attente JMS et en recevant des messages de ces dernières.

Utilisation de clients externes :

Un client externe (c'est-à-dire extérieur au serveur) peut avoir besoin d'interagir avec une application installée sur le serveur.

Le scénario présenté ici est très simple : un client externe souhaite interagir avec une application sur le serveur. La figure représente un scénario simple typique.

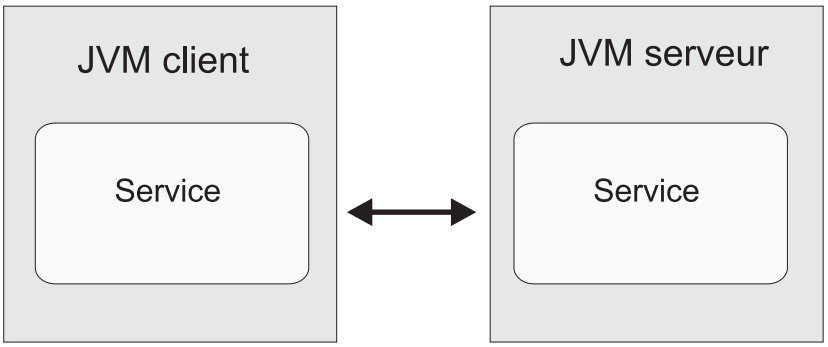


Figure 34. Scénario simple : interaction entre un client externe et une application du serveur

L'application SCA inclut une exportation avec une liaison JMS ; c'est ce qui la rend disponible aux clients externes.

Si un client externe se trouve sur une machine Java virtuelle (JVM) distincte de votre serveur, vous devez effectuer plusieurs opérations afin d'établir la connexion et permettre l'interaction avec une exportation JMS. Le client obtient un contexte initial (InitialContext) avec les valeurs appropriées, puis il compulse les ressources via JNDI. Le client utilise alors le client de spécification JMS 1.1 pour accéder aux destinations et envoyer/recevoir des messages sur les destinations.

Les noms JNDI par défaut des ressources créées automatiquement par l'environnement d'exécution sont répertoriés dans la rubrique configuration de cette section. Toutefois, si vous avez pré-créé les ressources, utilisez ces noms JNDI.

1. Configurer les destinations JMS et la fabrique de connexions pour envoyer le message.
2. Vérifier que le contexte JNDI, le port de l'adaptateur de ressources SIB et le port d'amorçage de la messagerie sont corrects.

   Le serveur utilise certains ports par défaut, mais si plusieurs serveurs sont installés sur ce système, des ports de remplacement sont créés lors de l'installation pour éviter les conflits avec d'autres instances de serveur. Vous pouvez utiliser la console d'administration pour déterminer quels ports votre serveur employer. Accédez à **Serveurs** > **Serveurs d'applications** > ***nom_serveur*** > **Configuration** et cliquez sur **Ports** sous **Communication**. Vous pouvez maintenant modifier le port utilisé.
3. Le client obtient un contexte initial avec les valeurs appropriées, puis il compulse les ressources via JNDI.
4. A l'aide des spécifications JMS 1.1, le client accède aux destinations et aux messages envoyés et reçus sur les destinations.

Identification et résolution des incidents liés aux liaisons JMS :

Vous pouvez diagnostiquer et résoudre des incidents survenant sur les liaisons JMS.

Exceptions liées à l'implémentation

En réponse à diverses conditions d'erreur, l'implémentation de l'importation et de l'exportation JMS peut renvoyer deux types d'exception :

- Service : cette exception est renvoyée si l'erreur définie sur l'interface métier de service (type de port WSDL) s'est produite.
- Service Runtime : générée dans tous les autres cas. Dans la plupart des cas, l'exception cause contient l'exception d'origine (JMSException).

  Par exemple, une importation n'attend qu'un seul message de réponse pour chaque message de demande. Si plusieurs réponses sont fournies, ou si une réponse tardive (une pour laquelle la réponse

SCA a expiré) est fournie, une exception Service Runtime est générée. La transaction est annulée et le message de réponse est retiré de la file d'attente ou traité par le gestionnaire d'événements ayant échoué.

Conditions d'erreur principales

Les principales conditions d'erreur liées aux liaisons JMS sont déterminées par la sémantique transactionnelle, la configuration du fournisseur JMS ou une référence au fonctionnement existant dans d'autres composants. Les causes premières d'incident peuvent être :

- Erreur de connexion au fournisseur ou à la destination JMS.

  Une erreur de connexion au fournisseur JMS pour recevoir des messages entraîne l'échec de démarrage de l'écouteur des messages. Cette situation sera enregistrée dans la journal WebSphere Application Server. Des messages persistants resteront sur la destination jusqu'à ce qu'ils soient récupérés (ou qu'ils parviennent à échéance).

  Une erreur de connexion au fournisseur JMS destinée à envoyer des messages entraîne l'annulation de la transaction qui contrôle l'envoi.
- Erreur d'analyse d'un message entrant ou de construction d'un message sortant.

  Une erreur de liaison des données ou du gestionnaire de données entraîne l'annulation de la transaction qui contrôle le travail.
- Erreur d'envoi du message sortant.

  Un échec d'envoi d'un message entraîne l'annulation de la transaction en question.
- Messages de réponse multiples ou inattendus.

  L'importation n'attend qu'un seul message de réponse pour chaque message de demande. Egalement la période valide durant laquelle une réponse peut être reçue par le qualifiant d'expiration de réponse SCA sur la demande. Lorsqu'une réponse est reçue ou le délai d'expiration expire, l'enregistrement de corrélation est supprimé. Si des messages de réponse parviennent de manière inattendue ou tardive, une exception Service Runtime est générée.
- Exception d'exécution lié au délai d'expiration du service, générée par une réponse tardive lors de l'utilisation du schéma de corrélation des destinations de réponse dynamique temporaires.

  L'importation JMS arrive à expiration après un délai déterminé par le qualifiant d'expiration des réponses SCA ; si ce délai n'est pas défini, il est de 60 secondes par défaut.

Messages SCA basés sur JM qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si les messages SCA émis par le biais d'une interaction JMS échouent, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Or, si ces messages n'apparaissent pas, vérifiez que la destination SIB sous-jacente de la destination JMS possède une valeur du nombre maximal de livraisons ayant échoué supérieure à **1**. Définir cette valeur sur **2** ou plus permet une interaction avec le gestionnaire des événements ayant échoué au cours des appels SCA pour les liaisons JMS.

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS, par défaut, est stocké dans le gestionnaire des événements ayant échoué pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Liaisons JMS génériques :

La liaison JMS générique assure la connectivité avec les fournisseurs tiers compatibles avec JMS 1.1. Le fonctionnement des liaisons JMS génériques est identique à celui des liaisons JMS.

Le service fourni par le biais d'une liaison JMS permet à un module SCA (Service Component Architecture) d'effectuer des appels ou de recevoir des messages à partir de systèmes externes. Le système peut être un système JMS externe.

La liaison JMS générique assure l'intégration avec les fournisseurs JMS conformes non JCA 1.5 qui prennent en charge JJMS 1.1 et implémentent la fonction de serveur d'applications JMS disponible en option. La liaison JMS générique prend en charge les fournisseurs JMS (notamment Oracle AQ, TIBCO, SonicMQ, WebMethods et BEA WebLogic) qui ne prennent pas en charge JCA 1.5 mais qui prennent en charge la fonction de serveur d'applications de la spécification JMS 1.1. Le fournisseur JMS imbriqué dans WebSphere (SIBJMS), qui est un fournisseur JMS JCA 1.5, n'est pas pris en charge par cette liaison ; si vous utilisez ce fournisseur, utilisez les «Liaisons JMS», à la page 118.

Cette liaison générique peut par exemple être utilisée lors d'une intégration avec un système JMS conforme non JCA 1.5 dans un environnement. Les applications externes cibles peuvent alors recevoir et envoyer des messages à intégrer avec un composant SCA.

Présentation des liaisons JMS génériques :

Les liaisons JMS génériques sont des liaisons JMS non JCA qui assurent la connectivité entre l'environnement SCA (Service Component Architecture) et les systèmes JMS compatibles avec JMS 1.1 et qui implémentent la fonction de serveur d'applications JMS en option.

Liaisons JMS génériques

Les aspects majeurs des liaisons d'importation et d'exportation JMS génériques sont les suivants :

- Port de l'écouteur : permet aux fournisseurs JMS non JCA de recevoir des messages et de les distribuer à un bean géré par message
- Connexions : encapsulent une connexion virtuelle entre un client et une application fournisseur
- Destinations : utilisées par un client pour spécifier la cible des messages produits ou la source des messages utilisés
- Données d'authentification : permettent de sécuriser l'accès à la liaison

Liaisons d'importation JMS génériques

Les liaisons d'importation JMS génériques permettent aux composants au sein de votre module SCA de communiquer avec les services fournis par les fournisseurs JMS conformes non JCA 1.5.

La connexion qui fait partie d'une importation JMS est une fabrique de connexions. Une fabrique de connexion, l'objet utilisé par un client pour créer une connexion à un fournisseur, encapsule un ensemble de paramètres de configuration de la connexion définie par un administrateur. Chaque fabrique de connexions est une instance de l'interface ConnectionFactory, QueueConnectionFactory ou TopicConnectionFactory.

L'interaction avec les systèmes JMS externes comprend l'utilisation des destinations pour l'envoi des requêtes et la réception des réponses.

Deux types de scénarios d'utilisation pour la liaison d'import ation JMS générique sont pris en charge, en fonction du type d'opération appelée :

- Unidirectionnel : l'importation JMS générique place un message sur la destination d'envoi configurée dans la liaison d'importation. Rien n'est envoyé à la zone replyTo de l'en-tête JMS.
- Bidirectionnel (demande-réponse) : l'importation JMS générique place un message sur la destination d'envoi et conserve ensuite la réponse reçue depuis le composant SCA.

  La destination receive est définie dans la propriété d'en-tête replyTo du message sortant. Un bean géré par message (MDB) est déployé pour écouter sur la destination de réception et, dès qu'une réponse est reçue, le MDB transmet la réponse au composant.

  La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut) ou à partir de l'ID de corrélation de message de demande.

Pour les scénarios d'utilisation unidirectionnel et bidirectionnel, les propriétés d'en-tête dynamique et statique peuvent être spécifiées. Les propriétés statiques peuvent être définies à partir de la liaison de méthode d'importation JMS générique. Certaines de ces propriétés revêtent des significations particulières pour l'environnement d'exécution JMS SCA.

Il est important de noter que la liaison JMS générique est une liaison asynchrone. Si un composant appelant appelle une importation JMS générique de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service JMS.

La figure 35, à la page 129 illustre la manière dont l'importation est liée au service externe.

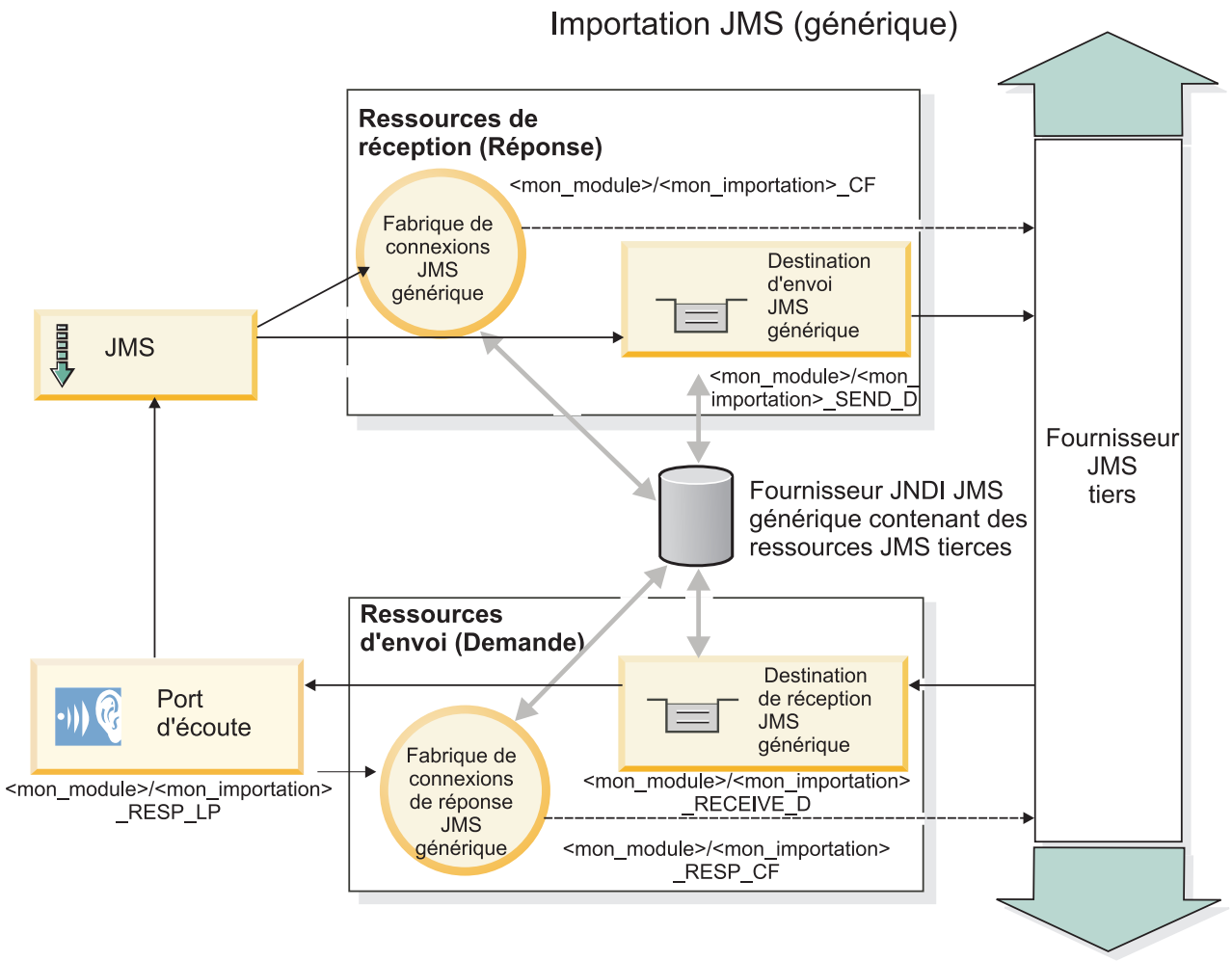


Figure 35. Ressources de liaisons d'importation JMS génériques

Liaisons d'exportation JMS génériques

Les liaisons d'exportation JMS génériques offrent les moyens aux modules SCA de fournir des services aux applications JMS externes.

La connexion qui fait partie d'une exportation JMS est composée d'une entité ConnectionFactory et d'une entité ListenerPort.

Une exportation JMS générique comporte des destinations d'envoi et de réception.

- La destination receive est le lieu de réception du message entrant destiné au composant cible.
- La destination send est celle à laquelle la réponse sera envoyée, sauf si le message entrant l'a remplacé en utilisant la propriété d'en-tête replyTo.

Un bean MDB est déployé pour écouter les demandes parvenant à la destination receive spécifiée dans la liaison d'exportation.

- La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse.
- La destination spécifiée dans la zone replyTo du message entrant remplace la destination spécifiée dans la zone send.
- Pour les scénarios de demande/réponse, la liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que la réponse

copie l'ID de message de demande dans la zone ID de corrélation du message de réponse (option par défaut) ou la réponse peut copier l'ID de corrélation de la demande dans la zone ID de corrélation du message de réponse.

La figure 36 illustre la manière dont le demandeur externe est lié à l'exportation.

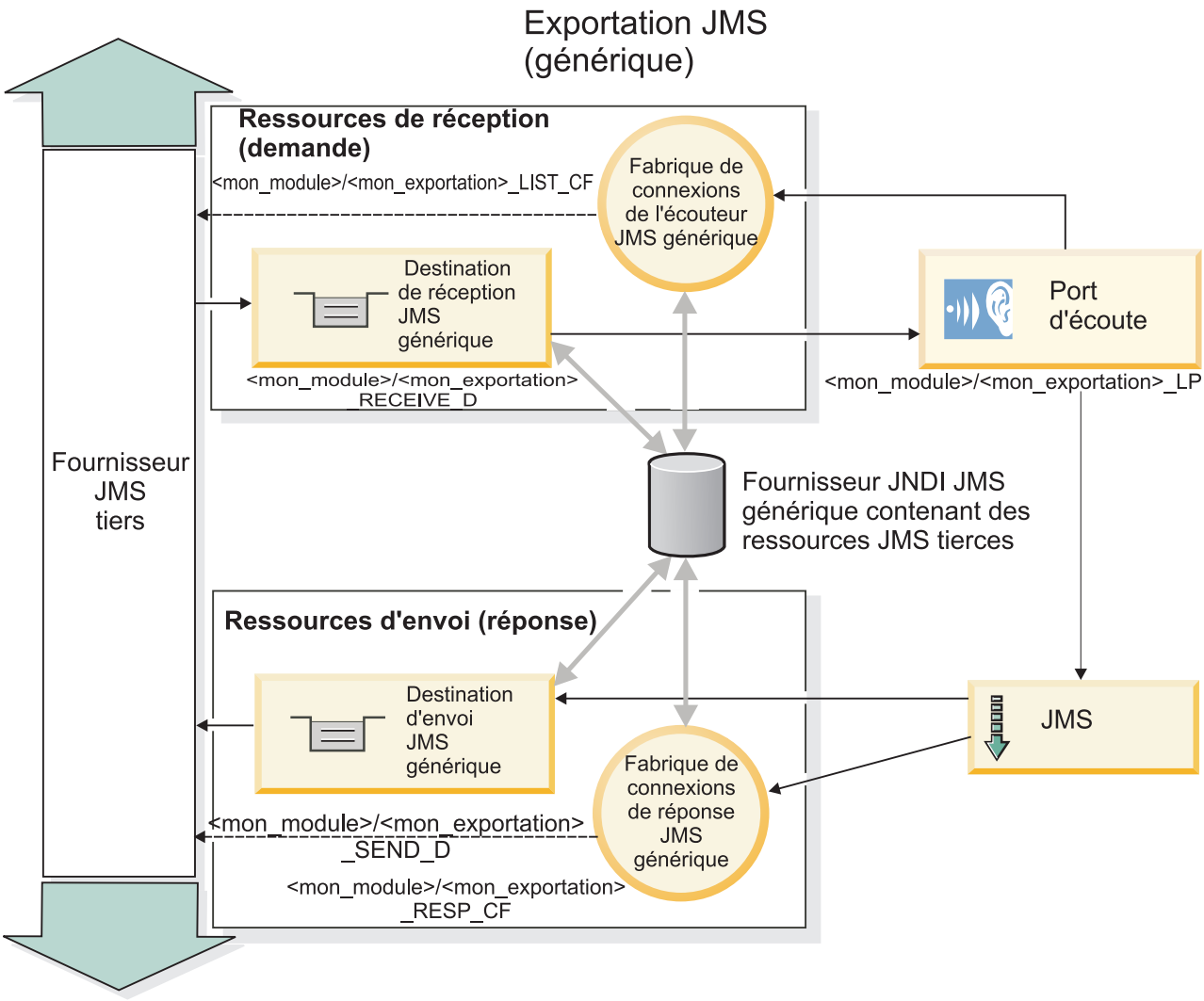


Figure 36. Ressources de liaisons d'exportation JMS génériques

Principales fonctionnalités des liaisons JMS Generic :

Les fonctions de la liaison Generic d'importation et d'exportation JMS correspondent aux fonctions des liaisons d'importation JMS et MQ JMS intégrées à WebSphere. Les fonctions essentielles incluent les définitions d'en-tête et l'accès aux ressources Java EE existantes. Cependant, étant donné son caractère générique, cette liaison ne comporte pas d'options de connectivité spécifiques au fournisseur JMS et elle est limitée à la génération de ressources lors du déploiement et de l'installation.

Importations génériques

Comme l'application d'importation MQ JMS, l'implémentation JMS Generic est asynchrone et prend en charge trois types d'appel : unidirectionnel, bidirectionnel (également appelé demande-réponse) et rappel.

Si l'importation JMS est déployée, un bean géré par message (MDB) fourni par l'environnement d'exécution est déployé. Le bean MDB est à l'écoute des réponses au message de demande. Le bean MDB est associé à (écoute) la destination envoyée avec la demande dans la zone d'en-tête replyTo du message JMS.

Exportations génériques

Les liaisons d'exportation JMS génériques diffèrent des liaisons d'exportation EIS au niveau du traitement du renvoi du résultat. Une exportation JMS Generic envoie explicitement la réponse à la destination replyTo spécifiée dans le message entrant. Si aucune destination n'est indiquée, la destination d'envoi est utilisée.

Lorsque l'exportation JMS Generic est déployée, un bean MDB (autre que le bean MDB utilisé pour les importations JMS Generic) est déployé. Il écoute les requêtes entrantes sur la destination de réception, puis distribue les requêtes que doit traiter l'environnement d'exécution SCA.

En-têtes spéciaux

Des propriétés d'en-tête spéciales sont utilisées dans les importations et les exportations JMS Generic pour indiquer à la cible comment traiter le message.

Par exemple, la propriété TargetFunctionName permet au sélecteur de fonction par défaut d'identifier le nom de l'opération sur l'interface d'exportation appelée.

Remarque : Il est possible de configurer la liaison d'importation pour que l'en-tête TargetFunctionName porte le nom de l'opération, dans chaque cas.

Ressources Java EE

Plusieurs ressources Java EE sont créées lors du déploiement d'une liaison JMS dans un environnement Java EE.

- Port d'écoute sur la destination de réception (réponse) (bidirectionnel uniquement) pour les importations, et sur la destination de réception (demande) pour les exportations
- Fabrique de connexions JMS générique pour outboundConnection (importation) et inboundConnection (exportation)
- Destination JMS générique pour les destinations d'envoi (importation) et de réception (exportation) (bidirectionnel uniquement)
- Fabrique de connexions JMS générique pour responseConnection (bidirectionnel uniquement et facultatif), sinon : outboundConnection est utilisé pour les importations et inboundConnection est utilisé pour les exportations)
- Destination JMS générique pour la destination de réception (importation) et d'envoi (exportation) (bidirectionnel uniquement)
- Destination JMS de rappel de fournisseur de messagerie par défaut utilisée pour l'accès à la destination de file d'attente de rappel SIB (bidirectionnel uniquement)
- Fabrique de connexions JMS de rappel de fournisseur de messagerie par défaut utilisée pour l'accès à la destination JMS de rappel (bidirectionnel uniquement)
- Destination de file d'attente de rappel SIB utilisée pour stocker les informations sur le message de demande à utiliser lors du traitement de la réponse (bidirectionnel uniquement)

La tâche d'installation crée ConnectionFactory, les trois destinations et ActivationSpec à partir des informations des fichiers d'importation et d'exportation.

En-têtes JMS génériques :

Les en-têtes JMS génériques sont des objets SDO (Service Data Objects) qui contiennent toutes les propriétés des propriétés de message JMS génériques. Ces propriétés peuvent provenir du message entrant ou il peut s'agir des propriétés qui seront appliquées au message sortant.

Un message JMS contient deux types d'en-têtes : l'en-tête de système JMS et plusieurs propriétés JMS. Il est possible d'accéder aux deux types d'en-tête depuis un module de médiation dans l'objet SMO (Service Message Object) ou en utilisant l'API ContextService.

Les propriétés suivantes sont définies statiquement sur methodBinding :

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

La liaison JMS générique JMS prend également en charge la modification dynamique des propriétés et des en-têtes JMS de la même manière que les liaisons JMS et JMS MQ.

Certains fournisseurs JMS génériques instaurent des restrictions sur quelles propriétés pouvant être définies par l'application et avec quelles combinaisons. Pour plus d'informations, reportez-vous à la documentation de votre produit tiers. Toutefois, une propriété complémentaire a été ajoutée à methodBinding, ignoreInvalidOutboundJMSProperties, qui permet de propager n'importe quelle exception.

Les propriétés de message et d'en-tête JMS génériques sont uniquement utilisées lorsque le commutateur de liaisons SCDL de l'architecture de composants de service de base est sous tension. Lorsque le commutateur est activé, les informations contextuelles sont propagées. Par défaut, ce commutateur est sous tension. Pour éviter la diffusion des informations contextuelles, remplacez la valeur par **false**.

Lorsque la diffusion du contexte est activée, les informations d'en-tête peuvent être transmises au message ou au composant cible. Pour activer et désactiver la diffusion du contexte, spécifiez la valeur **true** ou **false** pour l'attribut contextPropagationEnabled des liaisons d'importation et d'exportation. Exemple :

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

La valeur par défaut est **true**.

Résolution d'incidents liés aux liaisons JMS génériques :

Vous pouvez diagnostiquer et résoudre des incidents survenant sur les liaisons JMS génériques.

Exceptions liées à l'implémentation

En réponse à diverses conditions d'erreur, l'implémentation de l'importation et de l'exportation JMS générique peut renvoyer deux types d'exception :

- Service : cette exception est renvoyée si l'erreur définie sur l'interface métier de service (type de port WSDL) s'est produite.
- Service Runtime : générée dans tous les autres cas. Dans la plupart des cas, l'exception cause contient l'exception d'origine (JMSException).

Résolution d'une expiration de message JMS générique

Expiration d'un message de demande du fournisseur JMS.

L'*expiration de la demande* désigne l'expiration d'un message de demande du fournisseur JMS à la fin du délai JMSExpiration indiqué dans le message de demande. Comme pour d'autres liaisons JMS, la liaison JMS générique traite l'expiration de la demande en attribuant à l'expiration du message de rappel placé lors de l'importation la même valeur que celle de la demande sortante. La notification de l'expiration du message de rappel indique que le message de demande a expiré et le client doit être informé au moyen d'une exception.

Cependant, si la destination de rappel est transférée sur le fournisseur tiers, ce type d'expiration de demande n'est pas pris en charge.

L'*expiration de la réponse* désigne l'expiration d'un message de réponse du fournisseur JMS à la fin du délai JMSExpiration indiqué dans le message de réponse.

L'expiration de la réponse n'est pas prise en charge pour la liaison JMS générique, car le fonctionnement d'une expiration sur un fournisseur JMS tiers n'est pas définie. Toutefois, vous pouvez déterminer si la réponse a expiré lors de sa réception.

Pour les messages de demande sortants, la valeur de JMSExpiration est calculée à partir du temps d'attente et des valeurs requestExpiration indiquées dans asyncHeader, si elles sont définies.

Résolution d'erreurs liées aux fabriques de connexion JMS génériques

Lorsque vous définissez certains types de fabriques de connexion au niveau du fournisseur JMS générique, vous pouvez recevoir un message d'erreur lors d'une tentative de lancement d'une application. Vous pouvez modifier la fabrique de connexions externe pour empêcher ce problème de se produire.

Lorsque vous lancez une application, le message suivant peut s'afficher :

```
Le type JMSConnectionFactory du port d'écoute MDB ne correspond pas
au type JMSDestination
```

Cet incident peut survenir lorsque vous définissez des fabriques de connexions externes. En particulier, l'exception peut être générée lorsque vous créez une fabrique de connexions de sujet JMS 1.0.2 au lieu d'une fabrique de connexions JMS 1.1 (unifiée) (c'est-à-dire, une fabrique qui prenne en charge les communications de type point à point et publication/abonnement).

Pour résoudre ce problème, procédez comme suit :

1. Accédez au fournisseur JMS générique que vous utilisez.
2. Remplacez la fabrique de connexions de sujet JMS 1.0.2 que vous avez définie par une fabrique de connexions JMS 1.1 (unifiée).

Lorsque vous lancez l'application avec la nouvelle fabrique de connexions JMS 1.1, le message d'erreur ne devrait plus s'afficher.

Messages SCA génériques basés sur JMS qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si les messages SCA émis par le biais d'une interaction JMS générique échouent, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Si ces messages n'apparaissent pas, vérifiez que la valeur de la propriété du nombre maximal de nouvelles tentatives sur le port d'écoute sous-jacent est supérieure ou égale à 1. Définir cette valeur sur 1 ou plus permet une interaction avec le gestionnaire d'événements ayant échoué au cours des appels SCA pour les liaisons JMS génériques.

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS est stocké dans le gestionnaire des événements ayant échoué par défaut pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Le gestionnaire de données étant appelé par la liaison de données, une exception de gestionnaire de données est générée dans une exception de liaison de données. Par conséquent, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Liaisons JMS WebSphere MQ :

La liaison JMS WebSphere MQ assure l'intégration avec les applications externes qui utilisent un fournisseur basé sur JMS WebSphere MQ.

Utilisez les liaisons d'importation et d'exportationJMS WebSphere MQ lorsque vous souhaitez une intégration directe avec des systèmes JMS ou JMS MQ à partir de votre environnement de serveur. Ainsi, il n'est plus nécessaire d'utiliser les fonctions de lien MQ ou client du bus d'intégration de services.

Lorsqu'un composant interagit avec un service basé sur JMS WebSphere MQ par le biais d'une importation, la liaison d'importation JMS WebSphere MQ utilise une destination vers laquelle les données seront envoyées et une destination à laquelle la réponse peut être reçue. La conversion des données vers ou depuis un message JMS est accomplie via le composant Edge Component du gestionnaire de données ou de la liaison de données.

Lorsqu'un module SCA fournit un service aux clients JMS WebSphere MQ, la liaison d'exportation JMS WebSphere MQ utilise une destination à laquelle la requête peut être reçue et la réponse envoyée. La conversion des données vers et depuis un message JMS s'effectue par le biais du gestionnaire de données ou de la liaison de données JMS.

Le sélecteur de fonction sert à effectuer un mappage avec l'opération sur le composant cible à appeler.

Présentation des liaisons JMS WebSphere MQ :

La liaison JMS WebSphere MQ assure l'intégration aux applications externes qui utilisent le fournisseur JMS WebSphere MQ.

Tâches d'administration WebSphere MQ

Avant d'exécuter une application contenant ce type de liaison, l'administrateur système WebSphere MQ est censé créer le gestionnaire de files d'attente WebSphere MQ sous jacent qui sera utilisé par les liaisons JMS WebSphere MQ.

Liaisons d'importation JMS WebSphere MQ

L'importation JMS WebSphere MQ permet aux composants au sein de votre module SCA de communiquer avec des services offerts par les fournisseurs basés sur JMS WebSphere MQ. Vous devez utiliser une version prise en charge de WebSphere MQ. Pour plus de détails sur la configuration logicielle et matérielle requise, reportez-vous aux pages de support IBM..

Deux types de scénarios d'utilisation pour les liaisons d'importation JMS WebSphere MQ JMS sont pris en charge, en fonction du type d'opération appelée :

- Unidirectionnel : l'importation JMS WebSphere MQ place un message sur la destination d'envoi configurée dans la liaison d'importation. Rien n'est envoyé à la zone replyTo de l'en-tête JMS.
- Bidirectionnel (demande-réponse) : l'importation JMS WebSphere MQ place un message sur la destination d'envoi.

  La destination receive est définie dans la zone d'en-tête replyTo. Un bean géré par message (MDB) est déployé pour écouter sur la destination de réception et, dès qu'une réponse est reçue, le MDB transmet la réponse au composant.

  La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut) ou à partir de l'ID de corrélation de message de demande.

Pour les scénarios d'utilisation unidirectionnels et bidirectionnels, les propriétés d'en-tête dynamiques et statiques peuvent être spécifiées.Les propriétés statiques peuvent être définies à partir de la liaison de méthode d'importation JMS. Certaines de ces propriétés revêtent des significations particulières pour l'environnement d'exécution JMS SCA.

Il est important de noter que JMS WebSphere MQ est une liaison asynchrone. Si un composant appelant appelle une importation JMS WebSphere MQ de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service.

La figure 37, à la page 136 montre comment l'importation est liée au service externe.

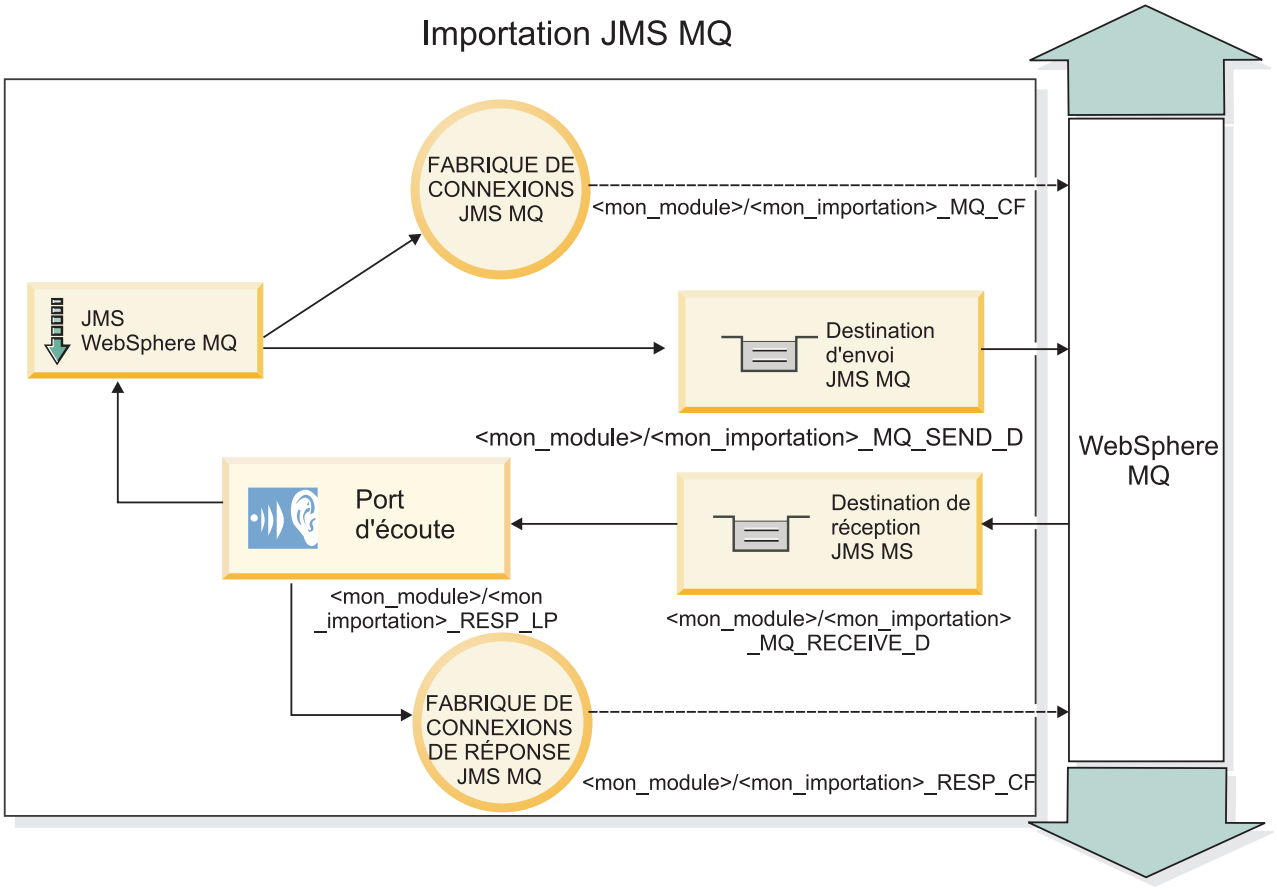


Figure 37. Ressources de liaison d'importation JMS WebSphere MQ

Liaisons d'exportation JMS Liaisons JMS WebSphere MQ

La liaison d'exportation JMS WebSphere MQ offre les moyens aux modules SCA de fournir des services aux applications JMS externes sur le fournisseur JMS basé sur WebSphere MQ.

Un bean MDB est déployé pour écouter les demandes parvenant à la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse. La destination spécifiée dans la zone replyTo du message de réponse remplace la destination spécifiée dans la zone send.

La figure 38, à la page 137 illustre la manière dont le demandeur externe est lié à l'exportation.

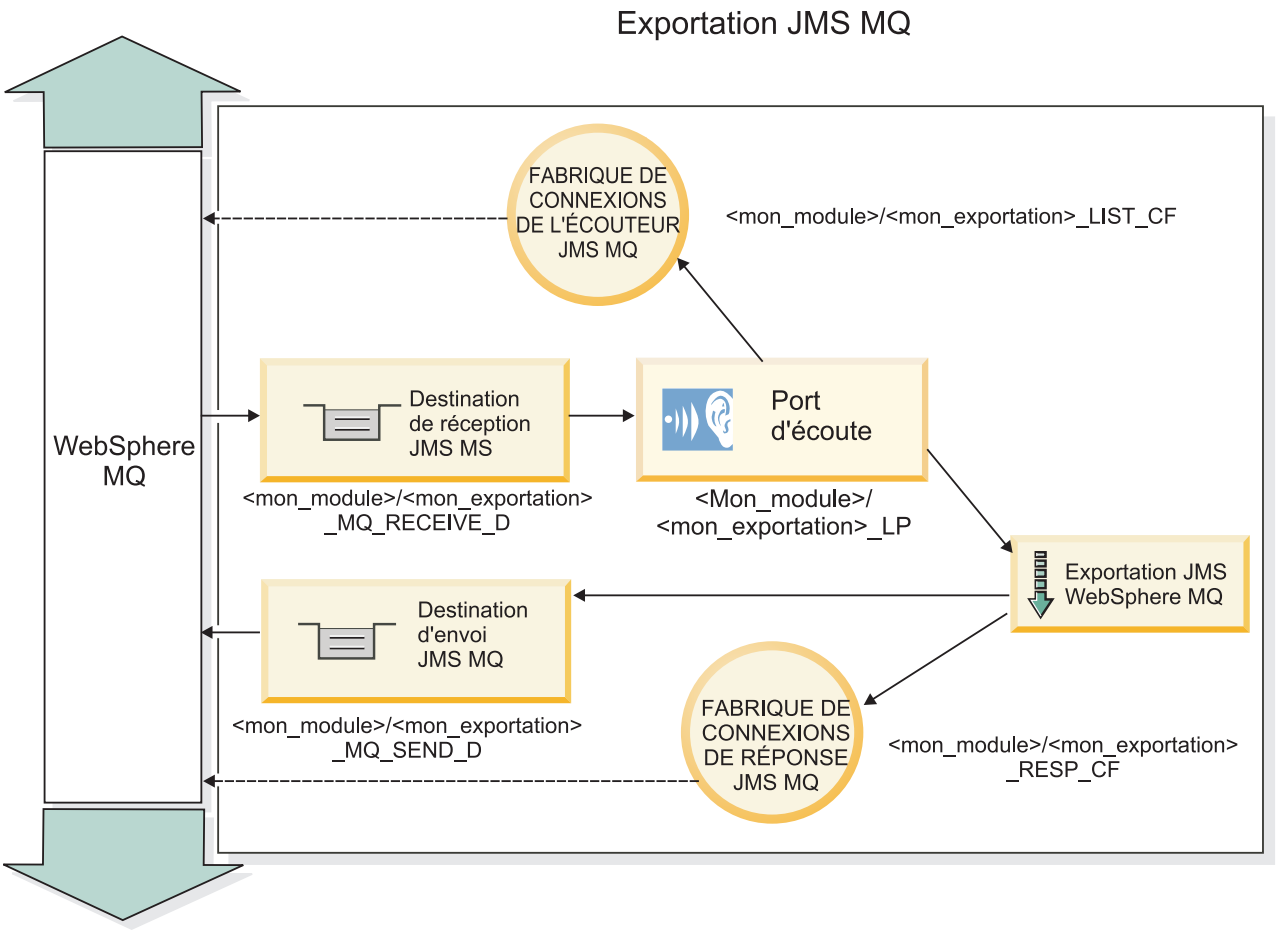


Figure 38. Ressources de liaison d'exportation JMS WebSphere MQ

Remarque : La figure 37, à la page 136 et la figure 38 illustrent comment une application d'une version précédente de IBM Business Process Manager est associée à un service externe. Pour les applications développées pour IBM Business Process Manager, version 7.0, la spécification d'activation est utilisée à la place du port de l'écouteur et de la fabrique de connexions.

Principales fonctionnalités des liaisons JMS WebSphere MQ :

Les fonctionnalités essentielles des liaisons JMS WebSphere MQ incluent les en-têtes, les artefacts Java EE et les ressources Java EE créées.

En-têtes

Un en-tête de message JMS contient plusieurs zones prédéfinies qui contiennent des valeurs utilisées par les clients et les fournisseurs pour identifier et acheminer les messages. Vous pouvez utiliser les propriétés de liaison pour configurer ces en-têtes avec des valeurs fixes, ou vous pouvez spécifier dynamiquement les en-têtes lors de l'exécution.

`JMSCorrelationID`
Liaison avec un message associé. En général, cette zone est définie sur la chaîne de l'identificateur du message auquel une réponse est envoyée.

`TargetFunctionName`
Cet en-tête est utilisé par l'un des sélecteurs de fonction fournis pour identifier l'opération appelée. La définition de la propriété d'en-tête JMS TargetFunctionName dans des messages envoyés à une exportation JMS permet l'utilisation de ce sélecteur de fonction. La propriété peut être définie

directement dans les applications client JMS ou lors de la connexion d'une importation avec une liaison JMS à une exportation de ce type. Dans ce cas, vous devez configurer la liaison d'importation JMS pour que l'en-tête TargetFunctionName pour chaque opération de l'interface soit défini sur le nom de l'opération.

Schémas de corrélation

Les liaisons JMS WebSphere MQ fournissent un grand nombre de schémas de corrélation qui permettent de déterminer la manière de corréler les messages de demande avec les messages de réponse.

RequestMsgIDToCorrelID
JMSMessageID est copié dans la zone JMSCorrelationID. Il s'agit du paramètre par défaut.

RequestCorrelIDToCorrelID
JMSCorrelationID est copié dans la zone JMSCorrelationID.

Ressources Java EE

Plusieurs ressources Java EE sont créées lors du déploiement d'une importation JMS MQ dans un environnement Java EE.

Paramètres

Fabrique de connexions MQ
Utilisée par les clients pour créer une connexion au fournisseur JMS MQ.

Fabrique de connexions de réponse
Utilisée par l'environnement d'exécution JMS MQ SCA lorsque la destination d'envoi se trouve sur un gestionnaire de files d'attente différent de celui de la destination de réception.

Spécification d'activation
Une spécification d'activation MQ JMS est associée à un ou plusieurs beans gérés par message et offre la configuration permettant de recevoir des messages.

Destinations

- Destination d'envoi :
  - Importations : Destination à laquelle la demande ou le message sortant est envoyé.
  - Exportations : Emplacement auquel est envoyé le message de réponse, si cette valeur n'est pas remplacée par l'en-tête JMSReplyTo dans le message entrant.
- Destination de réception :
  - Importations : Emplacement auquel est envoyé le message de réponse ou entrant.
  - Exportations : Emplacement dans lequel le message entrant doit être placé.

En-têtes JMS :

Un message JMS contient deux types d'en-têtes : l'en-tête de système JMS et plusieurs propriétés JMS. Il est possible d'accéder aux deux types d'en-tête depuis un module de médiation dans l'objet SMO (Service Message Object) ou en utilisant l'API ContextService.

En-tête système JMS

L'en-tête système JMS est représenté dans l'objet SMO par l'élément JMSHeader qui contient toutes les zones généralement présentes dans un en-tête JMS. Bien qu'elles puissent être modifiées dans la médiation (ou ContextService), certaines zones d'en-tête système JMS définies dans l'objet SMO ne seront pas propagées dans le message JMS sortant puisqu'elles sont remplacées par des valeurs statiques ou système.

Les zones clés de l'en-tête système JMS qui peuvent être mises à jour dans une médiation (ou ContextService) sont :

- **JMSType** et **JMSCorrelationID** : valeurs des propriétés de l'en-tête du message prédéfinies spécifiques
- **JMSDeliveryMode** : valeurs du mode de livraison (persistant (par défaut) ou non persistant)
- **JMSPriority** : valeur de la priorité (0 à 9 ; la valeur par défaut est JMS_Default_Priority)

Propriétés JMS

Les propriétés JMS sont représentées dans l'objet SMO en tant qu'entrées dans la liste Propriétés. Les propriétés peuvent être ajoutées, mises à jour ou supprimées dans une médiation ou en utilisant l'API ContextService.

Elles peuvent également être définies de manière statique dans la liaison JMS. Les propriétés définies de manière statique remplacent les paramètres (portant le même nom) qui sont définis de manière dynamique.

Les propriétés utilisateur propagées à partir d'autres liaisons (par exemple, une liaison HTTP) correspondront à une sortie dans la liaison JMS comme les propriétés JMS.

Paramètres de propagation d'en-tête

La propagation des propriétés et des en-têtes système JMS depuis le message JMS entrant vers les composants en aval ou depuis les composants en amont vers le message JMS sortant peut être contrôlée par l'indicateur Propagate Protocol Header.

Lorsque Propagate Protocol Header est défini, les informations d'en-tête peuvent circuler vers le message ou vers le composant cible, comme indiqué dans la liste suivante :

- Requête d'exportation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.
- Réponse d'exportation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS.
- Requête d'importation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS.
- Réponse d'importation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.

Clients externes :

Le serveur peut échanger des messages (envoi et réception) avec des clients externes par le biais de liaisons JMS WebSphere MQ.

Un client externe (comme un portail Web ou un système d'informations d'entreprise) peut envoyer un message à un composant SCA dans l'application par le biais d'une exportation ou il peut être appelé par un composant SCA dans l'application par le biais d'une importation.

La liaison d'exportation JMS WebSphere MQ déploie les beans MDB (Message Driven Beans) pour écouter les demandes entrantes sur la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si l'application appelée fournit une réponse. Ainsi, un client externe est en mesure d'appeler des applications via la liaison d'exportation.

Les importations JMS WebSphere MQ JMS effectuent des liaisons avec des clients externes et peuvent ainsi leur envoyer des messages. Ces messages peuvent ou non exiger une réponse de la part du client externe.

Pour plus d'informations sur l'interaction avec des clients externes à l'aide de WebSphere MQ, reportez-vous au centre de documentation WebSphere MQ.

Identification et résolution des incidents liés aux liaisons JMS WebSphere MQ :

Vous pouvez diagnostiquer et résoudre des incidents survenant sur les liaisons JMS WebSphere MQ.

Exceptions liées à l'implémentation

En réponse à diverses conditions d'erreur, l'implémentation de l'importation et de l'exportation JMS MQ peut renvoyer deux types d'exception :

- Service : cette exception est renvoyée si l'erreur définie sur l'interface métier de service (type de port WSDL) s'est produite.
- Service Runtime : générée dans tous les autres cas. Dans la plupart des cas, l'exception cause contient l'exception d'origine (JMSException).

  Par exemple, une importation n'attend qu'un seul message de réponse pour chaque message de demande. Si plusieurs réponses sont fournies, ou si une réponse tardive (une pour laquelle la réponse SCA a expiré) est fournie, une exception Service Runtime est générée. La transaction est annulée et le message de réponse est retiré de la file d'attente ou traité par le gestionnaire d'événements ayant échoué.

Messages SCA basés sur JMS WebSphere MQ qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si les messages SCA émis par le biais d'une interaction JMS WebSphere MQ échouent, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Or, si ces messages n'apparaissent pas, vérifiez que la valeur de la propriété du nombre maximal de nouvelles tentatives sur le port d'écoute sous-jacent est supérieure ou égale à **1**. Définir cette valeur sur **1** ou plus permet une interaction avec le gestionnaire d'événements ayant échoué au cours des appels SCA pour les liaisons JMS MQ.

Scénarios d'utilisation incorrecte : comparaison avec les liaisons WebSphere MQ

La liaison JMS WebSphere MQ est conçue pour l'interopérabilité avec les applications JMS déployées sur WebSphere MQ, où les messages affichés reposent sur un modèle de message JMS. Cependant, l'importation et l'exportationWebSphere MQ sont conçues essentiellement pour interopérer avec les applications WebSphere MQ natives et exposer le contenu intégral du corps de message WebSphere MQ aux médiations.

Dans les scénarios suivants, il est nécessaire d'utiliser la liaison JMSWebSphere MQ et non la liaison WebSphere MQ :

- Appel d'un bean géré par message (MDB) JMS à partir d'un module SCA, où le MDB est déployé sur le fournisseur JMS WebSphere MQ. Utilisez une importation JMS WebSphere MQ.
- Permettre au module SCA d'être appelé à partir d'un composant de servlet Java EE ou EJB par JMS. Utilisez une exportation JMS WebSphere MQ.
- Médiation du contenu d'un JMS MapMessage, transitant dans WebSphere MQ. Utilisez une exportation et une importation JMS WebSphere MQ conjointement avec le gestionnaire de données ou la liaison de données appropriés.

Dans certaines situations, la liaison WebSphere MQ et la liaison JMSWebSphere MQ peuvent interopérer. En particulier, si vous reliez des applications WebSphere MQ Java EE et non Java EE, utilisez une exportation WebSphere MQ et une importation JMS WebSphere MQ (ou vice-versa) conjointement avec les liaisons de données et/ou les modules de médiation appropriés (ou les deux).

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :
- Si le flux de médiation est configuré pour être transactionnel, le message JMS est stocké dans le gestionnaire des événements ayant échoué par défaut pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Le gestionnaire de données étant appelé par la liaison de données, une exception de gestionnaire de données est générée dans une exception de liaison de données. Par conséquent, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Liaisons WebSphere MQ :

La liaison WebSphere MQ assure une connectivité SCA (Service Component Architecture) avec les applications WebSphere MQ.

Utilisez les liaisons d'importation et d'exportation WebSphere MQ pour une intégration directe avec le système basé sur WebSphere MQ à partir de votre environnement serveur. Ainsi, il n'est plus nécessaire d'utiliser les fonctions de lien MQ ou client du bus d'intégration de services.

Lorsqu'un composant interagit avec un service WebSphere MQ par le biais d'une importation, la liaison d'importation WebSphere MQ utilise une file d'attente vers laquelle les données seront envoyées et une file d'attente où la réponse peut être reçue.

Lorsqu'un module SCA fournit un service aux clients WebSphere MQ, la liaison d'exportation WebSphere MQ utilise une file d'attente où la demande peut être reçue et la réponse envoyée. Le sélecteur de fonction sert à effectuer un mappage avec l'opération sur le composant cible à appeler.

La conversion des données utiles vers ou depuis un message MQ est effectuée par l'intermédiaire de la liaison de données ou du gestionnaire de données de corps MQ. La conversion des données d'en-tête vers et depuis un message MQ est effectuée par l'intermédiaire de la liaison de données d'en-tête MQ.

Pour plus d'informations sur les versions WebSphere MQ prises en charge, voir la page Web Configuration système requise détaillée.

Présentation des liaisons WebSphere MQ :

La liaison WebSphere MQ assure l'intégration avec les applications natives basées sur MQ.

Tâches d'administration WebSphere MQ

Avant d'exécuter une application contenant ce type de liaison, l'administrateur système WebSphere MQ est censé créer le gestionnaire de files d'attente WebSphere MQ sous jacent qui sera utilisé par les liaisons WebSphere MQ.

Tâches d'administration WebSphere

Vous devez spécifier comme propriété **Chemin d'accès aux bibliothèques natives**, de l'adaptateur de ressources MQ de Websphere, la version WebSphere MQ prise en charge par le serveur, puis redémarrer ce dernier. Cela garantit l'utilisation de bibliothèques d'une version de WebSphere MQ prise en charge. Pour plus de détails sur la configuration logicielle et matérielle requise, reportez-vous aux pages de support IBM..

Liaisons d'importation WebSphere MQ

La liaison d'importation WebSphere MQ permet aux composants au sein de votre module SCA de communiquer avec des services offerts par des applications externes basées sur WebSphere MQ. Vous devez utiliser une version prise en charge de WebSphere MQ. Pour plus de détails sur la configuration logicielle et matérielle requise, reportez-vous aux pages de support IBM..

L'interaction avec les systèmes WebSphere MQ externes comprend l'utilisation de file d'attentes pour l'envoi des requêtes et la réception des réponses.

Deux types descénarios d'utilisation pour la liaison d'importation WebSphere MQ sont pris en charge, en fonction du type d'opération appelée :

- Unidirectionnel : l'importation WebSphere MQ place un message sur la file d'attente configurée dans la zone **file d'attente de destination d'envoi** de la liaison d'importation. Rien n'est envoyé à la zone replyTo de l'en-tête MQMD.
- Bidirectionnel (demande-réponse) : l'importation WebSphere MQ place un message sur la file d'attente configurée dans la zone **file d'attente de destination d'envoi**

  La file d'attente receive est définie dans la zone d'en-tête MQMD replyTo. Un bean géré par message (MDB) est déployé pour écouter sur la file d'attente de réception et, dès qu'une réponse est reçue, le MDB transmet la réponse au composant.

La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse**) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut) ou à partir de l'ID de corrélation de message de demande.

Il est important de noter que WebSphere MQ est une liaison asynchrone. Si un composant appelant appelle une importation WebSphere MQ de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service WebSphere MQ.

La figure 39 montre comment l'importation est liée au service externe.

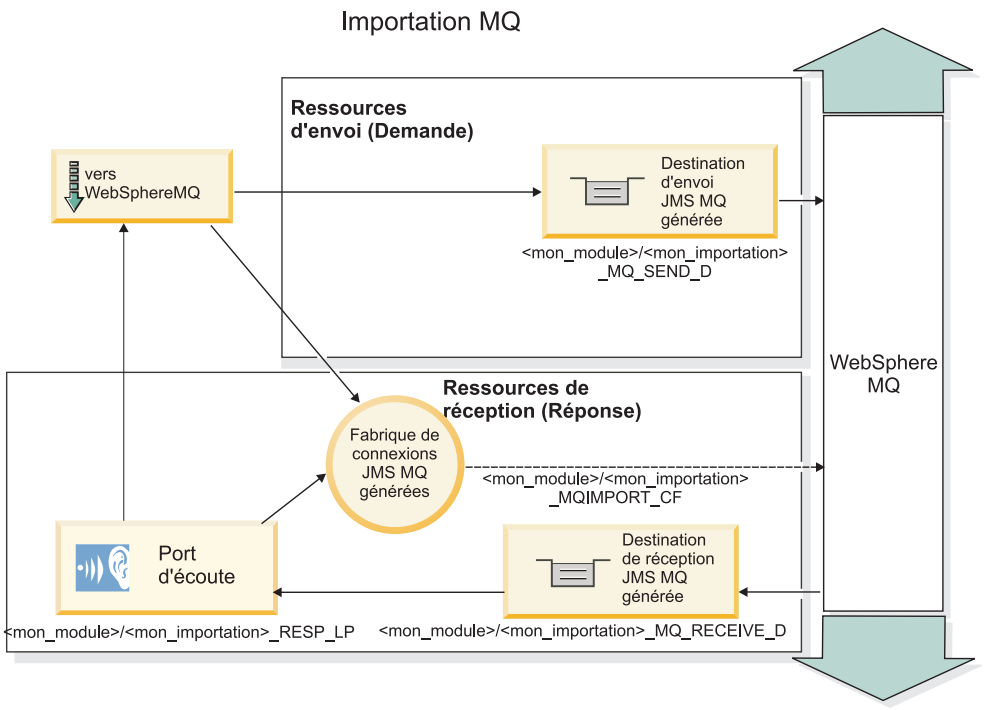


Figure 39. Ressources de liaison d'importation WebSphere MQ

Liaisons d'exportation WebSphere MQ

La liaison d'exportation WebSphere MQ offre les moyens aux modules SCA de fournir des services aux applications externes basées sur WebSphere MQ.

Un bean MDB est déployé pour écouter les demandes parvenant à la **file d'attente de destination de réception** spécifiée dans la liaison d'exportation. La file d'attente spécifiée dans la zone **file d'attente de destination d'envoi** est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse. La file d'attente spécifiée dans la zone replyTo du message de réponse remplace la file d'attente spécifiée dans la zone **file d'attente de destination d'envoi**.

La figure 40, à la page 144 illustre la manière dont le demandeur externe est lié à l'exportation.

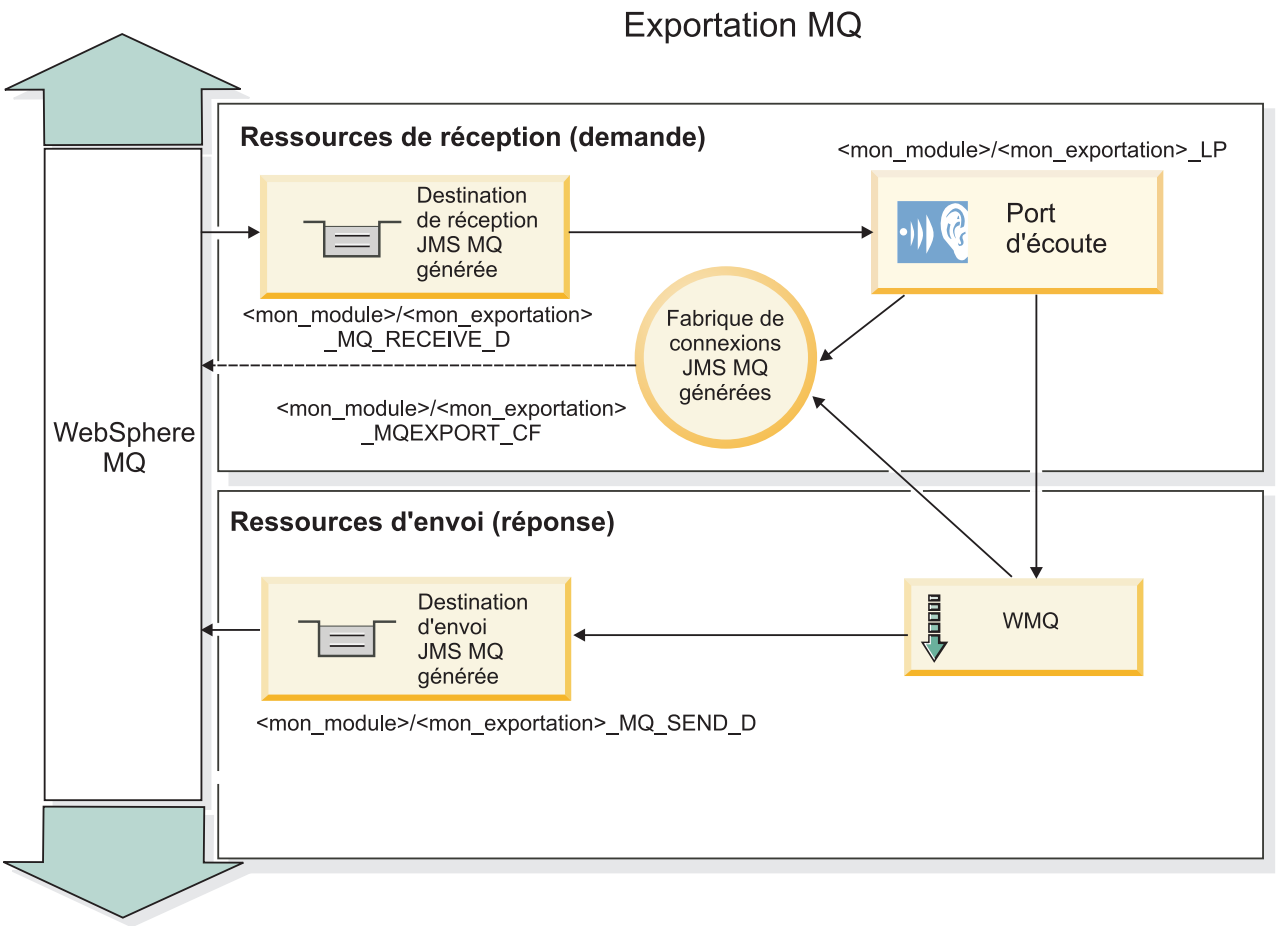


Figure 40. Ressources de liaison d'exportation WebSphere MQ

Remarque : La figure 39, à la page 143 et la figure 40 illustrent comment une application d'une version précédente de IBM Business Process Manager est associée à un service externe. Pour les applications développées pour IBM Business Process Manager, version 7 ou ultérieure, la spécification d'activation est utilisée à la place du port de l'écouteur et de la fabrique de connexions.

Principales fonctionnalités d'une liaison WebSphere MQ :

Les fonctionnalités essentielles d'une liaison WebSphere MQ incluent les en-têtes, les artefacts Java EE et les ressources Java EE créées.

Schémas de corrélation

Une application de demande/réponse WebSphere MQ peut utiliser différentes méthodes de corrélation entre les messages de réponse et les requêtes, qui reposent sur les zone MessageID et CorrelID du descripteur MQMD. Dans la plupart des cas, le demandeur laisse le gestionnaire de file d'attente sélectionner la valeur MessageID et s'attend à ce que l'application qui répond la copie dans la zone CorrelID de la réponse. En général, le demandeur et l'application qui répond savent implicitement quelle méthode de corrélation est utilisée. Parfois, l'application qui répond satisfait plusieurs indicateurs de la zone Report de la demande qui décrivent la méthode de traitement des zones.

Les liaisons d'exportation des messages WebSphere MQ peuvent être configurées à l'aide des options suivantes :

Options MsgId de la réponse :

New MsgID
Permet à un gestionnaire de file d'attente de sélectionner un MsgId pour la réponse (par défaut).

Copy from Request MsgID
Copie de la zone MsgId depuis la zone MsgId de la demande.

Copy from SCA message
Indique que le MsgId est celui des en-têtes de WebSphere MQ dans le message de réponse SCA, ou bien, en l'absence de valeur, c'est le gestionnaire de file d'attente qui définit un nouvel Id.

As Report Options
La zone Report du descripteur MQMD de la demande est analysée pour déterminer la manière de traiter MsgId. Les options MQRO_NEW_MSG_ID et MQRO_PASS_MSG_ID sont prises en charge et fonctionnent comme New MsgId et Copy from Request MsgID.

Options CorrelId de la réponse :

Copy from Request MsgID
Copie de la zone CorrelId depuis la zone MsgId de la demande (par défaut).

Copy from Request CorrelID
Copie de la zone CorrelId depuis la zone CorrelId de la demande.

Copy from SCA message
Indique que CorrelId est celui des en-têtes WebSphere MQ du message de réponse SCA, ou bien, en l'absence de valeur, cette zone reste vide.

As Report Options
La zone Report du descripteur MQMD de la demande est analysée pour déterminer la manière de traiter CorrelId. Les options MQRO_COPY_MSG_ID_TO_CORREL_ID et MQRO_PASS_CORREL_ID sont prises en charge et fonctionnent comme Copy from Request MsgID et Copy from Request CorrelID.

Les liaisons d'exportation des messages WebSphere MQ peuvent être configurées à l'aide des options suivantes :

Options MsgId de la demande :

New MsgID
Permet à un gestionnaire de file d'attente de sélectionner un MsgId pour la demande (par défaut).

Copy from SCA message
Indique que le MsgId est celui des en-têtes de WebSphere MQ dans le message de requête SCA, ou bien, en l'absence de valeur, permet au gestionnaire de file d'attente de définir un nouvel Id.

Options de corrélation de réponse :

Response has CorrelID copied from MsgId
Dans le message de réponse, la zone CorrelId doit être définie en fonction de la valeur MsgId de la requête (par défaut).

Response has MsgID copied from MsgId
Dans le message de réponse, la zone MsgId doit être définie en fonction de la valeur MsgId de la requête.

Response has CorrelID copied from CorrelId
Dans le message de réponse, la zone CorrelId doit être définie en fonction de la valeur CorrelId de la requête.

Ressources Java EE

Plusieurs ressources Java EE sont créées lorsqu'une liaison WebSphere MQ est déployée dans un environnement Java EE.

Paramètres

Fabrique de connexions MQ
Utilisée par les clients pour créer une connexion au fournisseur WebSphere MQ.

Fabrique de connexions de réponse
Utilisée par l'environnement d'exécution JMS MQ SCA lorsque la destination d'envoi se trouve sur un gestionnaire de files d'attente différent de celui de la destination d'envoi.

Spécification d'activation
Une spécification d'activation MQ JMS est associée à un ou plusieurs beans gérés par message et offre la configuration permettant de recevoir des messages.

Destinations

- Destination d'envoi : Destination à laquelle est envoyé le message de demande ou sortant (importation) ou le message de réponse (exportation), si la valeur n'est pas remplacée par la zone d'en-tête ReplyTo du MQMD du message entrant.
- Destination de réception : Emplacement dans lequel le message de réponse/demande ou le message entrant doit être placé.

En-têtes WebSphere MQ :

Les en-têtes WebSphere MQ incorporent certaines conventions pour la conversion aux messages de l'architecture SCA (Service Component Architecture).

Les messages WebSphere MQ se composent d'un en-tête système (le MQMD), de zéro ou plusieurs autres en-têtes MQ (système ou personnalisés) et d'un corps. S'il contient plusieurs en-têtes, leur ordre est important.

Chaque en-tête contient des informations décrivant la structure de l'en-tête suivant. Le MQMD décrit le premier en-tête.

Mode d'analyse des en-têtes MQ

Une liaison de données d'en-tête MQ est utilisée pour analyser les en-têtes MQ. Les en-têtes suivants sont automatiquement pris en charge :

- MQRFH
- MQRFH2
- MQCIH
- MQIIH

Les en-têtes qui commencent par **MQH** sont traités de façon différente. Certains champs de l'en-tête ne sont pas analysés ; ils persistent sous la forme d'octets non analysés.

Pour d'autres en-têtes MQ, vous pouvez écrire des liaisons de données d'en-tête MQ personnalisées afin d'analyser ces en-têtes.

Mode d'accès aux en-têtes MQ

Vous pouvez accéder aux en-têtes MQ de l'une des manières suivantes :

- Via l'objet de message de service (SMO) dans une médiation

- Via l'API ContextService

Les en-têtes MQ sont représentés en interne avec l'élément SMO MQHeader. MQHeader est un conteneur de données d'en-tête qui étend MQControl, mais qui contient un élément de valeur anyType. Il inclut MQMD, MQControl (informations de contrôle du corps de message MQ), ainsi qu'une liste d'autres en-têtes MQ.

- MQMD représente le contenu de la description du message WebSphere MQ, excepté pour les informations déterminant la structure et le codage du corps.
- MQControl contient des informations déterminant la structure et le codage d'un corps de message.
- MQHeaders contient une liste d'objets MQHeader.

La chaîne MQ n'est pas affectée de sorte que, dans le SMO, chaque en-tête MQ inclut ses propres informations de contrôle (CCSID, codage et format). Les en-têtes peuvent être facilement ajoutés ou supprimés, sans modifier d'autres données d'en-tête.

Définition de champs dans le MQMD

Vous pouvez mettre à jour le MQMD à l'aide de l'API Context ou via l'objet de message de service (SMO) dans une médiation. Les champs suivants sont automatiquement propagés vers le message MQ sortant :

- Encoding
- CodedCharacterSet
- Format
- Report
- Expiry
- Feedback
- Priority
- Persistence
- CorrelId
- MsgFlags

Configurez la liaison MQ sur une importation ou une exportation pour propager les propriétés suivantes vers le message MQ sortant :

MsgID
Définissez **Request Message ID** à `copy from SCA message`.

MsgType
Décochez la case **Set message type to MQMT_DATAGRAM or MQMT_REQUEST for request-response operation**.

ReplyToQ
Décochez la case **Override reply to queue of request message**.

ReplyToQMgr
Décochez la case **Override reply to queue of request message**.

A partir de la version 7.0, les champsde contexte peuvent être remplacés à l'aide d'une propriété personnalisée dans la définition de la destination JNDI. Définissez la propriété personnalisée MDCTX avec la valeur `SET_IDENTITY_CONTEXT` dans la destination d'envoi pour propager les champssuivants vers le message MQ sortant :

- UserIdentifier
- AppIdentityData

Définissez la propriété personnalisée MDCTX avec la valeur SET_ALL_CONTEXT dans la destination d'envoi pour propager les propriétés suivantes vers le message MQ sortant :

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Certains champs ne sont pas propagés vers le message MQ sortant. Les champs suivants sont remplacés lors de l'envoi du message :

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

Ajout statique de MQCIH dans une liaison WebSphere MQ :

IBM Business Process Manager prend en charge l'ajout des informations d'en-tête MQCIH statiquement sans utiliser de module de médiation.

Il existe différentes façons d'ajouter des informations d'en-tête MQCIH à un message (par exemple, en utilisant la primitive de médiation Configurateur d'en-tête). Il peut être utile d'ajouter des informations d'en-tête statiquement, sans utiliser de module de médiation supplémentaire. Les informations d'en-tête statique, y compris le nom du programme CICS, l'ID de transaction et d'autres détails d'en-tête de format de données peuvent être définis et ajoutés dans le cadre de la liaison WebSphere MQ.

WebSphere MQ, MQ CICS Bridge, et CICS doivent être configurés de telle sorte que les informations d'en-tête MQCIH soient ajoutées statiquement.

Vous pouvez utiliser Integration Designer de façon à configurer l'importation WebSphere MQ avec des valeurs statiques qui sont nécessaires pour les informations d'en-tête MQCIH.

Lorsqu'un message arrive et est traité par l'importation WebSphere MQ, une vérification est effectuée pour vérifier que les informations d'en-tête MQCIH figurent déjà dans le message. Si MQCIH est présent, les valeurs statiques définies dans l'importation WebSphere MQ sont utilisées pour remplacer les valeurs dynamiques correspondantes du message. Si MQCIH est absent, il en est créé un dans le message et les valeurs statiques définies dans l'importation WebSphere MQ sont ajoutées.

Les valeurs statiques définies dans l'importation WebSphere MQ sont spécifiques à une méthode. Vous pouvez spécifier différentes valeurs MQCIH statiques pour différentes méthodes au sein d'une même importation WebSphere MQ.

Cette fonction n'est pas utilisée pour donner des valeurs par défaut si MQCIH ne contient pas les informations d'en-tête spécifiques car une valeur statique définie dans l'importation WebSphere MQ remplacera la valeur correspondante fournie par le message entrant.

Clients externes :

IBM Business Process Manager peut échanger des messages (envoi et réception) avec des clients externes par le biais de liaisons WebSphere MQ.

Un client externe (comme un portail Web ou un système d'informations d'entreprise) peut envoyer un message à un composant SCA dans l'application par le biais d'une exportation ou il peut être appelé par un composant SCA dans l'application par le biais d'une importation.

La liaison d'exportation WebSphere MQ déploie les beans MDB (Message Driven Beans) pour écouter les demandes entrantes sur la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si l'application appelée fournit une réponse. Ainsi, un client externe est en mesure d'appeler des applications par l'intermédiaire d'une liaison d'exportation.

Les importations JMS WebSphere MQ effectuent des liaisons avec des clients externes et peuvent ainsi leur envoyer des messages. Ces messages peuvent ou non exiger une réponse de la part du client externe.

Pour plus d'informations sur l'interaction avec des clients externes à l'aide de WebSphere MQ, reportez-vous au centre de documentation WebSphere MQ.

Identification des incidents liés aux liaisons WebSphere MQ :

Diagnostic et correction des incidents et des erreurs liés aux liaisons WebSphere MQ.

Conditions d'erreur principales

Les principales conditions d'erreur liées aux liaisons WebSphere MQ peuvent être identifiées via la sémantique transactionnelle, la configuration de WebSphere MQ ou par référence au fonctionnement sur d'autres composants. Les causes premières d'incident peuvent être :

- Erreur de connexion au gestionnaire de file d'attente ou à la file d'attente WebSphere MQ.

  Une erreur de connexion à WebSphere MQ destinée à recevoir des messages entraîne l'échec de démarrage du port d'écoute MDB. Cette situation sera enregistrée dans la journal WebSphere Application Server. Des messages persistants resteront dans la file d'attente de WebSphere MQ jusqu'à ce qu'ils soient récupérés (ou que WebSphere MQ les fasse expirer).

  Une erreur de connexion à WebSphere MQ destinée à envoyer des messages sortants entraîne l'annulation de la transaction qui contrôle l'envoi.
- Erreur d'analyse d'un message entrant ou de construction d'un message sortant.

  Une erreur de liaison des données ou du gestionnaire de données entraîne l'annulation de la transaction qui contrôle le travail.
- Erreur d'envoi du message sortant.

  Un échec d'envoi d'un message entraîne l'annulation de la transaction en question.
- Messages de réponse multiples ou inattendus.

  L'importation n'attend qu'un seul message de réponse pour chaque message de demande. Si plusieurs réponses sont fournies, ou si une réponse tardive (une pour laquelle la réponse SCA a expiré) est fournie, une exception Service Runtime est générée. La transaction est annulée et le message de réponse est retiré de la file d'attente ou traité par le gestionnaire d'événements ayant échoué.

Scénarios d'utilisation incorrecte : comparaison avec les liaisons WebSphere MQ

L'importation et l'exportation WebSphere MQ sont conçues essentiellement pour interopérer avec les applications natives WebSphere MQ et exposer le contenu intégral du corps de message WebSphere MQ aux médiations. Toutefois, la liaison JMS WebSphere MQ est conçue pour l'interopérabilité avec les applications JMS déployées sur WebSphere MQ, où les messages affichés reposent sur un modèle de message JMS.

Dans les scénarios suivants, il est nécessaire d'utiliser la liaison JMSWebSphere MQ et non la liaison WebSphere MQ :

- Appel d'un bean géré par message (MDB) JMS à partir d'un module SCA, où le MDB est déployé sur le fournisseur JMS WebSphere MQ. Utilisez une importation JMS WebSphere MQ.
- Permettre au module SCA d'être appelé à partir d'un composant de servlet Java EE ou EJB par JMS. Utilisez une exportation JMS WebSphere MQ.
- Médiation du contenu d'un JMS MapMessage, transitant dans WebSphere MQ. Utilisez une exportation et une importation JMS WebSphere MQ conjointement avec la liaison de données appropriée.

Dans certaines situations, la liaison WebSphere MQ et la liaison JMSWebSphere MQ peuvent interopérer. En particulier, si vous reliez des applications WebSphere MQ Java EE et non Java EE, utilisez une exportation WebSphere MQ et une importation JMS WebSphere MQ (ou vice-versa) conjointement avec les liaisons de données et/ou les modules de médiation appropriés (ou les deux).

Messages non délivrés

Si WebSphere MQ ne parvient pas envoyer un message à la destination prévue (en règle générale, suite à des erreurs de configuration), il envoie le message à une file d'attente de rebut.

Dans ce cas, il ajoute un en-tête de non-distribution au début du corps de message. Ce dernier indique les raisons de l'erreur, la destination d'origine, ainsi que d'autres informations.

Messages SCA basés sur MQ qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si des messages SCA ont été générés en raison d'un incident d'interaction WebSphere MQ, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Si ces messages ne s'affichent pas dans le gestionnaire des événements ayant échoué, vérifiez que la destination WebSphere MQ sous-jacente possède une valeur du nombre maximal de livraisons ayant échoué supérieure à 1. Spécifier une valeur supérieure ou égale à 2 permet une interaction avec le gestionnaire d'événements ayant échoué au cours des appels SCA pour les liaisons WebSphere MQ.

Les événements MQ ayant échoué sont lus à nouveau sur le gestionnaire de files d'attente incorrect.

Quand une fabrique de connexions prédéfinie est utilisée pour les connexions sortantes, les propriétés de connexion doivent correspondre à celles définies dans la spécification d'activation employée pour les connexions entrantes.

La fabrique de connexions prédéfinies est utilisée pour créer une connexion quand un événement ayant échoué est relu. Elle doit donc être configurée pour utiliser le même gestionnaire de files d'attente duquel le message a été reçu à l'origine.

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS est stocké dans le gestionnaire des événements ayant échoué par défaut pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Le gestionnaire de données étant appelé par la liaison de données, une exception de gestionnaire de données est générée dans une exception de liaison de données. Par conséquent, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Limitations des liaisons :

L'utilisation des liaisons connaît certaines limitations qui sont répertoriées ici.

Limitations de la liaison MQ :

L'utilisation de la liaison MQ connaît certaines limitations qui sont répertoriées ici.

Pas de distribution de messages par publication/abonnement

La méthode de distribution des messages par publication/abonnement n'est pas prise en charge par la liaison MQ, bien que WMQ la prenne en charge. Toutefois, la liaison JMS MQ prend en charge cette méthode de distribution.

Files d'attentes de réception partagées

Plusieurs liaisons d'exportation et d'importation WebSphere MQ attendent que tous les messages présents dans la file d'attente de réception configurée soient destinés à l'exportation ou l'importation. Les liaisons d'importation et d'exportation doivent être configurées en prenant en compte ce qui suit :

- Chaque importation MQ doit avoir une file d'attente de réception distincte, car les liaisons d'importation MQ supposent que tous les messages dans la file d'attente de réception sont des réponses à des demandes qu'elles ont envoyées. Si la file d'attente de réception est partagée par plusieurs importations, les réponses peuvent être reçues par la mauvaise importation et ne seront pas mises en corrélation avec le message de demande d'origine.
- Chaque exportation MQ doit avoir une file d'attente de réception distincte, sinon vous ne pouvez pas prévoir quelle exportation recevra un message de demande déterminé.
- Les importations et les exportations MQ peuvent désigner la même file d'attente d'envoi.

Limitations des liaisons JMS, JMS MQ et JMS génériques :

Les liaisons JMS et JMS MQ connaissent certaines limitations.

Implications de la génération de liaisons par défaut

Les limitations de l'utilisation de liaisons JMS, JMS MQ et JMS génériques sont présentées dans les sections suivantes :

- Implications de la génération de liaisons par défaut
- Schéma de corrélation des réponses
- Support bidirectionnel

Lorsque vous générez une liaison, plusieurs zones sont renseignées automatiquement par défaut, si vous ne choisissez pas d'entrer les valeurs vous-même. Par exemple, un nom de fabrique de connexions est créé automatiquement. Si vous savez que vous placerez votre application sur un serveur et que vous y accéderez à distance à l'aide d'un client, vous devez, lors de la création de la liaison, entrez des noms JNDI plutôt que d'utiliser les valeurs par défaut car vous souhaiterez certainement contrôler ces valeurs via la console d'administration lors de la phase d'exécution.

Toutefois, si vous avez accepté les valeurs par défaut, mais que vous découvrez par la suite que vous ne pouvez pas accéder à votre application à partir d'un client éloigné, vous pouvez utiliser la console d'administration pour définir explicitement la valeur de la fabrique de connexions. Recherchez la zone des noeuds finaux du fournisseur dans les paramètres de la fabrique de connexions et ajoutez une valeur telle que la suivante : <nomhôte_serveur>:7276 (si vous utilisez le numéro de port par défaut).

Schéma de corrélation des réponses

Si vous utilisez le schéma de corrélation des réponses CorrelationId à CorrelationId, permettant de corréler les messages dans une opération demande/réponse, le message doit contenir un ID corrélation dynamique.

Pour créer un ID corrélation dynamique dans un module de médiation à l'aide de l'éditeur de flux de médiation, ajoutez une primitive de médiation de Mappage avant l'importation avec la liaison JMS. Ouvrez l'éditeur de mappage. Les en-têtes connus de l'architecture des composants de service seront disponibles dans le message cible. Faites glisser-déposer une zone contenant un ID unique dans le message dans l'ID corrélation de l'en-tête JMS du message cible.

Support bidirectionnel

Seuls les caractères ASCII sont pris en charge pour les noms JNDI (Java Naming and Directory Interface) lors de l'exécution.

Files d'attentes de réception partagées

Plusieurs liaisons d'exportation et d'importation attendent que tous les messages présents dans la file d'attente de réception configurée soient destinés à l'exportation ou l'importation. Les liaisons d'importation et d'exportation doivent être configurées en prenant en compte ce qui suit :

- Chaque liaison d'importation doit avoir une file d'attente de réception distincte, car ces liaisons supposent que tous les messages dans la file d'attente de réception sont des réponses à des demandes qu'elles ont envoyées. Si la file d'attente de réception est partagée par plusieurs importations, les réponses peuvent être reçues par la mauvaise importation et ne seront pas mises en corrélation avec le message de demande d'origine.
- Chaque exportation doit avoir une file d'attente de réception distincte, sinon vous ne pouvez pas prévoir quelle exportation recevra un message de demande déterminé.
- Les importations et les exportations peuvent désigner la même file d'attente d'envoi.

Objets métier

Les développeurs de logiciels ont mis au point plusieurs modèles et infrastructures de programmation dans lesquels les *objets métier* offrent une représentation naturelle des données métier pour le traitement des applications.

D'une manière générale, ces objets métier :

- Sont définis conformément aux normes du secteur informatique
- Mappent de façon transparente des données aux tables de base de données ou aux systèmes d'information d'entreprise.
- Prennent en charge des protocoles d'appels
- Fournissent la base du modèle de programmation des données pour la programmation des applications

Process Designer et Integration Designer fournissent aux développeurs un modèle d'objet métier commun pour la représentation des divers types d'entités commerciales issues de domaines différents.

Dans Process Designer, les objets métier reposent sur une représentation de type de données. Des objets métier de base (types de variable) sont fournis dans des kits d'outils système. Vous pouvez créer des types de variable personnalisés appelés "objets métier personnalisés". Pour plus d'informations, reportez-vous à la rubrique Objets métier et variables.

Dans Integration Designer, disponible avec IBM Business Process Manager Advanced, les objets métier peuvent représenter des constructions XSD plus complexes. Dans Integration Designer, les objets métier ont une affinité étroite avec les schémas XML. Pour plus d'informations sur les éléments à prendre en considération lors de l'intégration des objets métier définis dans Integration Designer à des objets métier définis dans Process Designer, reportez-vous aux rubriques Library mirroring et Constructions XML non prises en charge.

Lors de la phase de développement dans Integration Designer, le modèle d'objet métier permet aux développeurs de définir des objets métier sous la forme de définitions de schémas XML. Lors de la phase d'exécution, les données métier établies par les définitions de schéma XML sont représentées sous forme d'objets métier Java. Dans ce modèle, les objets métier s'appuient assez librement sur des avant projets de la spécification SDO (Service Data Object) et offrent un jeu complet d'interfaces d'applications des modèles de programmation requis pour la manipulation des données de gestion. Les sous-rubriques suivantes expliquent plus en détail comment vous pouvez utiliser des objets métier avec Integration Designer.

Définir des objets métier

La définition d'objets métier s'effectue à l'aide de l'éditeur livré avec Integration Designer. Cet éditeur stocke les objets métier sous forme de définitions de schémas XML.

L'utilisation d'un schéma XML pour définir des objets métier présente plusieurs avantages :

- Les schémas XML fournissent un modèle standard de définition des données et une base pour l'interopérabilité entre des systèmes et des applications disparates et hétérogènes. Ils s'utilisent conjointement au langage WSDL (Web Services Description Language) pour fournir des contrats d'interface standard entre les composants, les applications et les systèmes.
- Les modèles de définition de données qu'ils produisent sont riches en informations permettant de représenter les données métier. Entre autres fonctionnalités intéressantes, ces modèles incluent des types complexes, des types simples, des types définis par l'utilisateur, l'héritage des types et la cardinalité.
- Les objets métier peuvent être définis aussi bien par les interfaces métier et les données définies dans le langage WSDL (Web Services Description Language) que par des schémas XML émanant d'organismes édicteurs de normes sectorielles ou provenant d'autres systèmes ou d'autres applications. Integration Designer peut importer directement ces objets métier.

Integration Designer permet également de détecter les données métier présentes dans des bases de données et dans des systèmes d'information d'entreprise et de générer les schémas XML de ces données métier. L'on désigne fréquemment les objets métier générés de la sorte sous le nom d'*objets métier propres à une application* car ils imitent la structure des données métier définies dans le système d'information d'entreprise.

Lorsqu'un processus manipule des données provenant d'un grand nombre de systèmes d'information différents, il peut être intéressant de transformer les représentations disparates des données métier (par exemple, CustomerEIS1 et CustomerEIS2 ou OrderEIS1 et OrderEIS2) en une représentation canonique unique (par exemple, Customer ou Order). L'on désigne souvent la représentation canonique sous le nom d'*objet métier générique*.

Les définitions d'objets métier, et c'est particulièrement vrai des objets métier génériques, sont souvent utilisées par plus d'une application. Pour permettre cette réutilisation, Integration Designer autorise la création d'objets métier dans des bandothèques qui peuvent dès lors être associées à plusieurs modules d'application.

Le langage WSDL (Web Services Description Language) définit les contrats des services fournis et utilisés par un module d'application SCA (Service Component Architecture), ainsi que les contrats permettant de créer les composants d'un module d'application. Dans un contrat, un WSDL peut représenter les opérations et les objets métier (définis par un schéma XML pour représenter les données métier).

Utilisation des objets métier

SCA (Service Component Architecture) fournit le cadre de travail permettant de définir un module d'application, les services que fournit ce module, ceux qu'il utilise et la composition des composants fournissant la logique métier du module. Les objets métier jouent un rôle important dans l'application, en définissant les données métier servant à décrire les contrats du service et de ses composants ainsi que les données métier manipulées par ces composants.

Le schéma suivant montre un module d'application SCA avec l'indication d'un grand nombre d'endroits où le développeur exploite les objets métier.

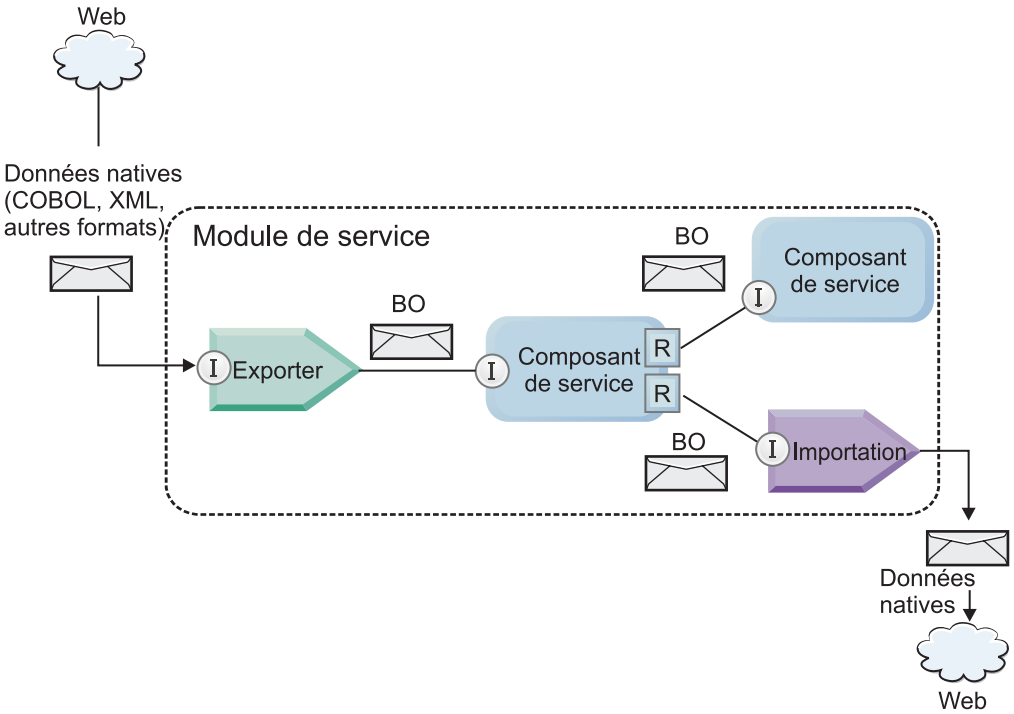


Figure 41. Les objets métier représentent les données qui circulent entre les services dans une application.

Remarque : Nous allons décrire l'utilisation des objets métier par les modules d'applications SCA. Si vous utilisez des interfaces Java, les modules d'applications SCA pourront également traiter les objets Java.

Modèle de programmation des objets métier

Le modèle de programmation des objets métier est constitué d'un ensemble d'interfaces Java représentant :

- la définition et les données d'instance des 'objets métier
- un ensemble de services prenant en charge les opérations sur les objets métier

Les définitions de type d'objet métier sont représentées par les interfaces commonj.sdo.Type et commonj.sdo.Property. Le modèle de programmation fournit un ensemble de règles pour le mappage vers l'interface Type des types complexes de schémas XML et pour le mappage de chacun des éléments de la définition de type complexe vers l'interface Property.

Les instances d'objets métier sont représentées par l'interface commonj.sdo.DataObject. Le modèle de programmation des objets métier est sans types, ce qui signifie que la même interface commonj.sdo.DataObject peut être utilisée pour représenter plusieurs définitions d'objets métier différentes, comme Customer et Order, par exemple. Savoir quelles propriétés seront définies et extraites de chaque objet métier dépend des informations de type définies dans le schéma XML associé à l'objet métier.

Le comportement du modèle de programmation d'objet métier repose sur la spécification Service Data Object 2.1. Pour plus d'informations, voir la spécification SDO 2.1 for Java, les tutoriels et les javadocs sur le site Web : http://www.oasis-opencsa.org/.

Les services d'objets métier prennent en charge diverses opérations liées au cycle de vie (création, égalité, analyse syntaxique, sérialisation, etc.) des objets métier.

Pour des informations détaillées sur le modèle de programmation des objets métier, voir Programmation à l'aide de services d'objet métier ainsi que la documentation sur les interfaces API et SPI générées relative aux objets métier.

Liaisons, liaisons de données et gestionnaires de données

Comme le montre la figure 41, à la page 154, les données métier servant à appeler des services fournis par les modules d'applications SCA sont transformées en objets métier pour permettre aux composants SCA de manipuler les données métier. De la même manière les objets métier manipulés par les composants SCA sont convertis dans le format de données requis par les services externes.

Dans certains cas, comme la liaison à un service Web, la liaison servant à exporter et importer des services transforme automatiquement les données dans le format approprié. Dans d'autres cas, comme dans la liaison JMS, les développeurs peuvent fournir une liaison de données ou un gestionnaire de données qui convertissent les formats non natifs en objets métier représentés par l'interface DataObject.

Pour plus d'informations sur le développement de liaisons et de gestionnaires de données, voir «Gestionnaires de données», à la page 58 et «Liaisons de données», à la page 60.

Composants

Les composants SCA définissent leurs contrats de services de fourniture et d'utilisation à l'aide d'une combinaison de langage WSDL et de schémas XML. Les données métier que SCA transmet entre les composants sont représentées sous forme d'objets métier à l'aide de l'interface DataObject. SCA vérifie que ces types d'objets métier sont compatibles avec le contrat d'interface défini par le composant à appeler.

Les abstractions du modèle de programmation pour la manipulation des objets métier varient d'un composant à l'autre. Le composant POJO et la primitive Custom du composant de flux de médiation

fournissent une manipulation directe des objets métier en permettant la programmation Java directe à l'aide des interfaces de programmation et des services d'objets métier. La plupart des composants fournissent des abstractions générales pour la manipulation des objets métier tout en fournissant également des fragments de code Java pour la définition de comportements personnalisés dans les interfaces et les services d'objets métier.

Il est possible de transformer les objets métier à l'aide des composants Interface Flow Mediation et Business Object Map (Mappe d'objet métier) ou du composant de flux de médiation et de sa primitive Mappe XML. Ces possibilités de transformations sont utiles pour la conversion dans les deux sens d'objets métier spécifiques à une application en objets métier génériques.

Objets métier spéciaux

Les objets de message de service et les graphiques métier sont deux types spécialisés d'objets métier qui sont utilisés à des fins spécifiques.

Objet de message de service

Un objet de message de service (SMO) est un objet métier spécialisé qui est utilisé par les composants de flux de médiation pour représenter la collection des données associées à un appel de service.

Un SMO a une structure fixe de premier niveau constituée des en-têtes, du contexte, du corps du message et des éventuelles pièces jointes.

- Les en-têtes charrient les informations relatives à l'appel de service via un protocole ou une liaison particulière. Exemples : en-têtes SOAP ou JMS.
- Les données de contexte transportent les informations logiques supplémentaires associées à l'appel lorsque celui-ci est traité par le composant de flux de médiation. En principe, ces informations ne font pas partie des données d'application envoyées ou reçues par les clients.
- Le corps du SMO contient les données métier elles-mêmes, qui représentent sous la forme d'un objet métier standard le message central de l'application ou les données de l'appel.

Le SMO peut également transporter des données jointes dans le cas d'appels à des services Web utilisant SOAP avec des pièces jointes.

Les flux de médiation effectuent des tâches comme le routage des demandes et la transformation des données et le SMO unifie dans une même structure les en-têtes et le contenu du message.

Graphique métier

Un graphique métier est un objet métier spécial servant à permettre la synchronisation des données dans des scénarios d'intégration.

Prenons l'exemple de deux systèmes d'information d'entreprise qui ont une représentation d'une commande spécifique. Lorsque la commande est modifiée dans l'un des deux systèmes, un message peut être envoyé à l'autre système afin de synchroniser les données de la commande. Les graphiques métier se chargent d'expédier à l'autre système uniquement la portion de la commande qui a changé, en ajoutant des annotations définissant précisément la teneur des modifications.

Dans notre exemple, un graphique métier Order n'enverra à l'autre système que l'article de la commande qui a été supprimé ainsi que la date modifiée de l'expédition de la commande.

Dans Integration Designer, il est facile d'ajouter des graphiques métier aux objets métier existants. Ils sont généralement utilisés dans les scénarios dans lesquels des adaptateurs WebSphere sont utilisés et pour prendre en charge la migration des applications WebSphere InterChange Server.

Mode d'analyse syntaxique d'objet métier

Integration Designer fournit une propriété pour les modules et les bibliothèques, qui permet de configurer le mode d'analyse XML pour les objets métier : rapide ou lent.

- Si le mode doit être *rapide*, les flux des octets XML sont analysés rapidement pour la création de l'objet métier.
- Avec l'option *lent*, l'objet métier est créé normalement et l'analyse effective du flux des octets XML est différée, une analyse partielle ne s'effectuant que lors de l'accès aux propriétés de l'objet métier.

Dans les deux modes, les données non XML sont toujours analysées de manière rapide pour la création de l'objet métier.

Remarques relatives au choix du mode d'analyse syntaxique des objets métier :

Le mode d'analyse syntaxique des objets métier détermine le mode d'analyse syntaxique des données XML. Un mode d'analyse syntaxique d'objet métier est défini dans un module ou une bibliothèque lors de sa création. Vous pouvez changer le mode d'analyse syntaxique du module ou de la bibliothèque, mais vous devez connaître les conséquences.

Le mode d'analyse d'objet métier est défini au niveau du module ou de la bibliothèque. Les modules créés dans une version IBM Integration Designer antérieure à la version 7 fonctionnent dans le mode d'analyse syntaxique systématique sans avoir à effectuer des modifications. Par défaut, les modules et les bibliothèques créés dans IBM Integration Designer version 7 et les versions suivantes sont affectés du mode d'analyse syntaxique le plus approprié en fonction de divers facteurs, tels que le mode d'analyse syntaxique des projets existants dans l'espace de travail ou du mode d'analyse syntaxique des projets dépendants ou d'autres projets dans la même solution, etc. Vous pouvez changer le mode d'analyse syntaxique du module ou de la bibliothèque en fonction de l'implémentation, mais vous devez connaître les conséquences.

Considérations

- Le mode d'analyse syntaxique lente des objets métier traite plus vite les données XML, mais il existe des différences de compatibilité que vous devez connaître entre le mode systématique et le mode lent avant de changer la configuration d'un module ou d'une bibliothèque. Ces différences affectent le comportement d'exécution des modules. Pour plus d'informations sur le mode d'analyse le plus efficace pour l'application, voir "Avantages du mode d'analyse lent par rapport au mode d'analyse syntaxique systématique.
- Un module peut être configuré pour être exécuté dans un des modes d'analyse syntaxique. Les bibliothèques peuvent être configurées pour prendre en charge l'un ou l'autre mode ou les deux modes. Une bibliothèque configurée pour prendre en charge les deux modes peuvent être référencée par un module utilisant le mode d'analyse syntaxique systématique et un module utilisant le mode d'analyse syntaxique lente. Le mode d'analyse syntaxique d'une bibliothèque lors de l'exécution est déterminé par les modules qui font référence à la bibliothèque. Lors de l'exécution, un module déclare son mode d'analyse syntaxique qui est utilisé par le module et les bibliothèques utilisés par le module.
- Les modules et les bibliothèques configurés pour différents modes d'analyse syntaxique sont compatibles dans les cas suivants :
  - Les modules et les bibliothèques configurés avec le mode d'analyse syntaxique lente sont compatibles avec les bibliothèques qui utilisent le mode d'analyse syntaxique lente ou les modes d'analyse syntaxique systématique et lente.
  - Les modules et les bibliothèques configurés avec le mode d'analyse syntaxique systématique sont compatibles avec les bibliothèques qui utilisent le mode d'analyse syntaxique systématique ou les modes d'analyse syntaxique systématique et lente.
  - Les bibliothèques configurées avec les modes d'analyse syntaxique systématique et lente sont compatibles uniquement avec les bibliothèques qui utilisent les modes d'analyse syntaxique systématique et lente.

- Utilisez le même mode d'analyse pour les modules qui interagissent qui communiquent en utilisant la liaison SCA. Si les modules communiquent en utilisant des modes d'analyse syntaxique différents, des problèmes de performance peuvent apparaître.

Concepts associés:

«Avantages comparés des deux modes d'analyse syntaxique : mode rapide et mode lent»
Certaines applications tirent avantage du mode passif d'analyse syntaxique du XML alors que d'autres voient leurs performances améliorées avec le mode attentif d'analyse. Il est fortement conseillé de tester l'application dans les deux modes afin de déterminer lequel convient le mieux aux caractéristiques spécifiques de l'application.

Avantages comparés des deux modes d'analyse syntaxique : mode rapide et mode lent :

Certaines applications tirent avantage du mode passif d'analyse syntaxique du XML alors que d'autres voient leurs performances améliorées avec le mode attentif d'analyse. Il est fortement conseillé de tester l'application dans les deux modes afin de déterminer lequel convient le mieux aux caractéristiques spécifiques de l'application.

Les applications qui analysent des flux importants de données XML constateront très probablement une amélioration de leurs performances si l'on utilise le mode lent. Ces améliorations se font d'autant plus sentir que la taille des flux d'octets XML augmente, alors que la quantité des données analysées par l'application dans ce flux diminue.

Les applications suivantes sont censées optimiser leurs performances dans ce mode :

- les applications qui analysent des flux de données non XML
- les applications utilisant des messages créés à l'aide du service BOFactory
- les applications qui analysent des messages XML de taille très modeste

Référence associée:

«Remarques relatives au choix du mode d'analyse syntaxique des objets métier», à la page 157
Le mode d'analyse syntaxique des objets métier détermine le mode d'analyse syntaxique des données XML. Un mode d'analyse syntaxique d'objet métier est défini dans un module ou une bibliothèque lors de sa création. Vous pouvez changer le mode d'analyse syntaxique du module ou de la bibliothèque, mais vous devez connaître les conséquences.

Remarques concernant la migration et le développement d'applications :

Si vous configurez une application développée à l'origine avec le mode d'analyse syntaxique rapide et voulez utiliser le mode lent, ou si vous prévoyez d'utiliser alternativement les deux modes sur une application donnée, vous devez tenir compte des différences existant entre ces deux modes et de ce qu'implique le passage d'un mode à l'autre.

Gestion des erreurs

Des exceptions d'analyse syntaxique se produisent si le flux d'octets XML est malformé.

- En mode d'analyse syntaxique XML rapide, ces exceptions apparaissent dès que l'objet métier est analysé dans le flux XML entrant.
- Si c'est le mode lent qui est configuré, les exceptions se produisent tardivement au moment non seulement de l'accès aux propriétés de l'objet métier, mais encore à celui où est analysée la portion malformée du XML.

Pour traiter le XML malformé, vous avez plusieurs possibilités :

- déployer aux extrémités un bus de service d'entreprise pour valider le XML entrant
- concevoir une logique de détection passive des erreurs au niveau du point où l'accès se fait aux propriétés des objets métier

Piles et messages d'exceptions

L'implémentation sous-jacente des deux modes étant différente, les traces des piles générées par les interfaces de programmation et les services des objets métier ont beau porter le même nom de classe d'exception, elles risquent de contenir des messages différents pour les exceptions ou d'encapsuler de manière différente les classes d'exceptions.

Format de sérialisation XML

Le mode lent fournit une optimisation des performances qui, lors de la sérialisation, tente de copier vers le flux d'octets sortant le XML non modifié à partir du flux d'octets entrant. Il en résulte indéniablement un surcroît de performances, mais le format de la sérialisation du flux d'octets XML sortant risque d'être différent si la totalité de l'objet métier a été actualisé en mode lent ou s'il s'exécutait en mode rapide.

Bien que le format de la sérialisation du XML risque de ne pas être précisément équivalent du point de vue de la syntaxe, la valeur sémantique apportée par l'objet métier, elle, est équivalente quel que soit le mode d'analyse, et le XML peut être passé en toute sécurité entre des applications s'exécutant dans des modes d'analyse syntaxique équivalents sur le plan sémantique.

Validateur d'instances d'objets métier

Le validateur d'instances d'objets métier en mode d'analyse XML lente fournit une validation plus fidèle des objets métier, particulièrement la validation des facettes des valeurs des propriétés. Du fait de ces plus, le validateur d'instances en mode lent intercepte des problèmes que ne voit pas le mode rapide et il fournit des messages d'erreur plus détaillés.

XML Maps version 602

Les flux de médiation originellement développés avant la version 6.1 de WebSphere Integration Developer risquent de contenir des primitives Mappage qui utilisent une mappe ou une feuille de styles ne pouvant pas s'exécuter directement en mode d'analyse syntaxique XML lent. Lorsqu'on fait migrer une application pour l'utiliser en mode passif, les fichiers de mappes associés aux primitives Mappage peuvent être automatiquement mis à jour. Par contre, si une primitive Mappage se réfère directement à une feuille de styles qui a été éditée manuellement, cette feuille de styles ne migre pas et elle ne peut pas s'exécuter en mode d'analyse syntaxique XML lent.

API privées non publiées

Si une application exploite des interfaces de programmation d'objets métier spécifiques à une implémentation, la compilation de l'application a toutes les chances d'échouer lorsqu'on change de mode d'analyse syntaxique. En mode rapide, ces interfaces privées sont normalement des classes d'implémentation d'objets métier qui sont définies par Eclipse Modeling Framework (EMF).

Dans tous les cas, il est fortement recommandé que les API privées soit retirées de l'application.

API EMF d'objets de message de service

Un composant de médiation de IBM Integration Designer permet de manipuler le contenu des message à l'aide des classes et des interfaces Java fournies dans le package com.ibm.websphere.sibx.smobo. En mode lent, les interfaces Java de ce package sont toujours utilisables mais les méthodes qui se réfèrent directement aux classes et interfaces EMF ou qui sont hérités des interfaces EMF ont toutes les chances d'échouer.

En mode lent, il est impossible de transtyper en objets EMF ServiceMessageObject et son contenu.

Service BOMode

Le service BOMode sert à déterminer quel est le mode d'analyse du XML en cours d'exécution : mode rapide ou mode lent.

Migration

Toutes les applications antérieures à la version 7.0.0.0 s'exécutent en mode rapide. Lorsque, au moment de l'exécution, on les fait migrer à l'aide des outils de migration de l'environnement d'exécution BPM, elles continuent à s'exécuter en mode rapide.

Pour permettre à une application antérieure à la version 7.0.0.0 d'utiliser le mode lent, vous devez préalablement utiliser Integration Designer pour faire migrer les artefacts de l'application. Après la migration, vous pouvez alors configurer l'application pour qu'elle utilise le mode lent.

Voir Migration d'artefacts source pour savoir comment faire migrer des artefacts dans Integration Designer ; voir également Configuration du mode d'analyse syntaxique des objets métier des modules et des bibliothèques pour savoir comment définir le mode d'analyse.

Relations

Une relation est une association entre deux ou plusieurs entités de données, généralement des objets métier. Dans IBM Business Process Manager Advanced, les relations permettent de transformer des données équivalentes contenues dans plusieurs objets métier et d'autres données représentées différemment. Elles peuvent également être utilisées pour créer des associations entre différents objets métier se trouvant dans plusieurs applications. Il est possible de les partager sur différents produits, applications et solutions.

Le service de relation de IBM Business Process Manager Advanced fournit l'infrastructure et les opérations permettant de gérer les relations. Grâce à ses capacités de gestion des objets métier où qu'ils se trouvent, ce service peut fournir une vue unifiée et globale sur toutes les applications d'une entreprise et servir de bloc fonctionnel pour les solutions BPM. Extensibles et gérables, les relations peuvent être utilisées dans des solutions d'intégration complexes.

Définition d'une relation

Une relation est une association entre plusieurs objets métier. Chaque objet métier faisant partie d'une relation est appelé *participant* de cette relation. Chaque participant se distingue des autres par sa fonction (ou *rôle*) au sein de cette relation. Une relation contient une liste de rôles.

Sa *définition* détaille chaque rôle et indique la façon dont ils sont reliés. Elle détaille également la "forme" globale de la relation. Par exemple, un rôle donné peut n'avoir qu'un seul participant alors qu'un autre peut en disposer de plusieurs. Par exemple, vous pouvez définir une relation *voiture-propriétaire*, dans laquelle un propriétaire peut posséder plusieurs voitures. Autre exemple, une instance peut avoir plusieurs participants pour chacun des rôles suivants :

- Voiture (Ferrari)
- Propriétaire (Jean-Marc)

La définition de la relation constitue un modèle pour l'*instance* de relation. L'instance constitue l'instanciation de l'exécution de la relation. Dans l'exemple *voiture-propriétaire*, elle peut décrire n'importe laquelle des associations suivantes :

- Jean-Marc possède une Ferrari
- Sarah possède une Mazda
- Robert possède une Ferrari

Les relations vous permettent d'éviter d'avoir à personnaliser la persistance de suivi des relations dans votre logique métier. Dans certains cas, le service de relation fait tout le travail à votre place (voir l'exemple de la section concernant les relations d'identité).

Scénarios

Voici un exemple classique de situation dans laquelle une solution d'intégration peut avoir recours aux relations. Un groupe important achète plusieurs entreprises (ou unités métier). Chacune utilise différents logiciels pour effectuer un suivi du personnel et des ordinateurs portables. L'entreprise cherche à surveiller ses employés et ses ordinateurs portables et recherche une solution lui permettant de :

- Visualiser tous les employés des différentes unités métier comme s'ils étaient regroupés dans une base de données unique
- Avoir une vue unique de tous ses ordinateurs portables
- Permettre à ses employés de se connecter au système et d'acheter un ordinateur portable
- Intégrer les différents systèmes d'applications d'entreprise au niveau des unités métier

Pour cela, le groupe doit trouver un moyen de s'assurer, par exemple, que Jean-Marc Dupond et Jean-M. Dupond, qui figurent dans différentes applications, soient vus comme un seul et même employé. Par exemple, le groupe recherche un moyen de consolider une entité unique sur plusieurs espaces d'applications.

Les scénarios de relations plus complexes impliquent la mise en place de processus BPEL qui créent des relations entre différents objets se trouvant sur plusieurs applications. Grâce à de tels scénarios, les objets métier sont regroupés dans la solution d'intégration et ne sont plus disséminés dans les applications. Le service de relation fournit une plateforme permettant de gérer les relations de manière persistante (auparavant, vous deviez créer votre propre service de persistance d'objet). Les deux exemples suivants constituent des scénarios de relation complexes :

- Un objet métier **voiture** disposant d'un numéro VIN se trouve dans une application SAP et vous souhaitez effectuer un suivi du fait que cette voiture appartient à quelqu'un d'autre. Toutefois, la relation de propriété concerne une personne référencée dans une application PeopleSoft. Dans ce cas, deux solutions sont impliquées et vous devez créer une passerelle pour les relier.
- Un important groupe de distribution veut pouvoir effectuer un suivi des marchandises renvoyées contre un avoir ou un remboursement. Deux applications différentes sont impliquées : un système de gestion des commandes (pour les achats) et un système de gestion des retours (pour les retours). Les objets métier se trouvent dans plusieurs applications et vous devez trouver un moyen pour afficher les relations entre eux.

Modèles d'utilisation courants

Les modèles d'utilisation les plus courants sont les modèles d'*équivalence*, qui fonctionnent sur une base de références croisées ou de corrélations. Deux types de relations correspondent à ce modèle : *non-identité* et *identité*.

- Les **relations de non-identité** établissent des associations entre des objets métier ou d'autres données sur une base un à plusieurs ou plusieurs à plusieurs. Dans chaque instance de relation, il peut y avoir une ou plusieurs instances de chaque participant. La relation de recherche statique est un type de relation de non-identité. Exemple : une relation entre l'objet **CA** d'une application SAP et l'élément **Californie** d'une application Siebel.
- Les **relations d'identité** établissent des associations entre les objets métier ou d'autres données sur une base un à un. Dans chaque instance de relation, il ne peut y avoir qu'une seule instance de chaque participant. Les relations d'identité capturent les références croisées entre les objets métier qui sont équivalents au niveau sémantique mais qui sont identifiés différemment sur plusieurs applications. Chaque participant de la relation est associé à un objet métier disposant d'une valeur (ou d'une

combinaison de valeurs) qui identifie l'objet de façon unique. Les relations d'identité transforment généralement les attributs clés des objets métier, comme les ID numériques et les codes produit.

Par exemple, si vous disposez d'objets métier **voiture** dans des applications SAP, PeopleSoft et Siebel et si vous recherchez une solution permettant de les synchroniser, vous devez intégrer manuellement une logique de synchronisation des relations en six mappes :

SAP -> générique

générique -> SAP

PeopleSoft-> générique

générique -> PeopleSoft

Siebel-> générique

générique -> Siebel

Mais, si vous utilisez des relations dans votre solution, le service de relation fournit des implémentations prédéfinies de modèles qui gèrent à votre place toutes ces instances de relations.

Outils de gestion des relations

L'*éditeur de relations* contenu dans Integration Designer est un outil permettant de concevoir les rôles et les relations d'intégration métier. Pour obtenir des informations détaillées et des informations sur la création de relations et l'utilisation de l'éditeur de relations, reportez-vous à la rubrique Création de relations.

Le *service de relation* est un service d'infrastructure de IBM Business Process Manager qui administre les relations et les rôles du système et exécute les opérations de gestion des rôles et des relations.

Le *gestionnaire de relations* constitue l'interface d'administration permettant la gestion des relations. Pour y accéder, utilisez les pages correspondantes de la console d'administration.

Il est possible d'appeler des relations automatiquement à l'aide des API de service de relation.

Service de relation

Ce service stocke les données de relation dans les tables de relation où il effectue un suivi des valeurs spécifiques à l'application sur plusieurs applications et solutions. Il permet des opérations de gestion des relations et des rôles.

Fonctionnement de la relation

Les relations et les rôles sont définis à l'aide de l'interface graphique de l'éditeur de relations de Integration Designer. Le service de relation stocke les données de corrélation dans les tables de la base de données de relation, dans la source de données par défaut que vous définissez lors de la configuration du service de relation. Une autre table (parfois appelée "table de participant") stocke les informations de chaque participant de la relation. Le service de relation utilise ces tables pour effectuer un suivi des valeurs spécifiques à l'application correspondante et pour propager les informations mises à jour sur l'intégralité des solutions.

Les relations, qui sont des artefacts métier, sont déployées sur un projet ou une bibliothèque partagée. Au premier déploiement, le service de relation intègre les données.

Au moment de l'exécution, lorsque les mappes ou autres composants IBM Business Process Manager requièrent une instance de relation, les instances de la relation sont extraites ou mises à jour, selon le scénario choisi.

Les données d'instance de relation et de rôle peuvent être manipulées de trois façons :

- Appels des API du service de relation par des snippets Java du composant IBM Business Process Manager

- Transformations de relations dans le service de mappage d'objets métier de IBM Business Process Manager
- Utilisation de l'outil de gestion de relations

Pour obtenir des informations détaillées et des informations sur la création de relations et l'utilisation de l'éditeur de relations, l'identification des types de relations et l'utilisation de l'éditeur de relations, reportez-vous à la rubrique Création de relations.

Gestionnaire de relations

Le gestionnaire de relations constitue l'interface d'administration permettant la gestion des relations. Pour y accéder, utilisez les pages correspondantes de la console d'administration.

Le gestionnaire de relations offre une interface utilisateur graphique permettant de créer et de manipuler les données de rôles et de relations au moment de l'exécution. Vous pouvez gérer les entités de relation à tous les niveaux : instance de relation, instance de rôle, données d'attribut et données de propriété. Le gestionnaire de relations vous permet :

- D'afficher la liste des relations du système ainsi que des informations détaillées concernant les relations individuelles
- De gérer les instances de relation :
  - D'interroger les données de relation pour afficher les sous-ensembles de données d'instance
  - D'interroger les données de relation pour afficher les sous-ensembles de données d'instance à l'aide des vues de base de données
  - D'afficher la liste des instances de relation correspondant à une requête de relation ainsi que des informations détaillées concernant une instance
  - De modifier les valeurs de propriété d'une instance de relation
  - De créer et de supprimer des instances de relation
- De gérer les rôles et les instances de rôle :
  - D'afficher des informations détaillées concernant un rôle ou une instance de rôle
  - De modifier les propriétés d'une instance de rôle
  - De créer et de supprimer des instances de rôle correspondant à une relation
  - De restaurer des données d'instance de relation à un moment donné quand vous êtes sûr de la fiabilité de ces données
- D'importer des données d'une relation statique existante dans votre système ou d'exporter ces données vers un fichier RI ou CSV.
- De supprimer le modèle et les donnés d'une relation à partir du référentiel lors de la désinstallation de l'application qui l'utilise

Relations en environnement de déploiement réseau

Il est possible d'utiliser les environnements de déploiement réseau (ND) sans configuration supplémentaire.

En environnement de déploiement réseau (ND), les relations sont installées dans un cluster d'applications. Elles sont visibles dans ce cluster et tous les serveurs qui s'y trouvent ont alors accès aux données d'instance stockées dans la base de données de relations. La possibilité d'exécuter le service de relation dans un environnement ND rend ce dernier évolutif et hautement disponible.

Le gestionnaire de relations permet la gestion de relations sur plusieurs clusters via une interface d'administration centralisée. Pour connecter le gestionnaire de relations à un serveur faisant partie d'un cluster, sélectionnez sa relation MBean.

API de service de relation

Il est possible d'appeler des relations automatiquement à l'aide des API de service de relation, à partir ou hors des mappes d'objets métier.

Trois types d'API sont disponibles :

- API de manipulation d'instance de relation (dont création, mise à jour et suppression directe de données d'instance)
- API de prise en charge de patterns de relation (dont correlate(), correlateforeignKeyLookup)
- patterns de recherche de relation (API de recherche)

Bus de service d'entreprise de IBM Business Process Manager

IBM Business Process Manager prend en charge l'intégration des services d'application avec les mêmes fonctions que WebSphere Enterprise Service Bus.

Connexion de services via un bus de service d'entreprise

Avec un bus de service d'entreprise (ESB), vous pouvez optimiser la souplesse d'une architecture SOA. Les participants d'une interaction de service sont connectés à l'ESB, plutôt que directement à un autre module.

Quand le demandeur de services se connecte à l'ESB, l'ESB est responsable de la transmission de ses demandes, à l'aide de messages, au fournisseur de services proposant la fonction et la qualité de service requises. L'ESB simplifie les interactions demandeur-fournisseur et s'occupe de la non concordance des protocoles, des patterns d'interaction ou des fonctions de service. Un ESB peut également activer ou améliorer le contrôle et la gestion. L'ESB offre des fonctions de gestion et de virtualisation qui implémentent et étendent les principales fonctionnalités de l'architecture SOA.

L'ESB extrait les fonctions suivantes :

Emplacement et identité
: Les participants n'ont pas besoin de connaître l'emplacement ou l'identité des autres participants. Par exemple, les demandeurs n'ont pas besoin de savoir qu'une demande peut être traitée par n'importe lequel des nombreux fournisseurs ; les fournisseurs de services peuvent être ajoutés ou supprimés sans perturbation.

Protocole d'interaction
: Les participants n'ont pas besoin de partager le même protocole de communication ou le même style d'interaction. Par exemple, une demande exprimée en tant que SOAP via HTTP peut être gérée par un fournisseur comprenant uniquement SOAP via Java Message Service (JMS).

Interface
: Les demandeurs et les fournisseurs n'ont pas besoin de s'entendre sur une interface commune. Un ESB synchronise les différences en convertissant les messages de demande et de réponse dans un format attendu par le fournisseur.

Qualités de (interaction) service
: Les participants, ou administrateurs système, expriment leurs exigences en termes de qualité de service, notamment l'autorisation des demandes, le chiffrement et déchiffrement du contenu des messages, l'audit automatique des interactions de service, ainsi que l'acheminement souhaité de leur demandes (privilégiant la rapidité ou le coût, par exemple).

L'interposition de l'ESB entre les participants vous permet de moduler leurs interactions via une construction logique appelée *médiation*. Les médiations agissent sur les messages en cours entre les demandeurs et le fournisseurs. Par exemple, les médiations permettent de trouver des services avec des caractéristiques spécifiques recherchées par un demandeur, ou de résoudre des différences d'interface entre demandeurs et fournisseurs. Pour les interactions complexes, les médiations peuvent être reliées successivement.

A l'aide des médiations, un bus de service d'entreprise exécute les actions suivantes entre le demandeur et le service :

- *Acheminement* des messages entre les services. Un bus de service d'entreprise offre une infrastructure de communication commune permettant de se connecter aux services et donc aux fonctions métier qu'ils représentent, sans que les programmeurs aient à écrire et gérer une logique de connectivité complexe.
- *Conversion* des protocoles de transport entre le demandeur et le service. Un bus de service d'entreprise est un moyen cohérent normalisé d'intégrer des fonctions métier qui utilisent des normes informatiques différentes. Cela permet de disposer de fonctions métier qui ne pourraient normalement pas communiquer, telles que la connexion d'applications dans des silos départementaux ou la participation des applications de différentes sociétés aux interactions de service.
- *Conversion* des formats de message entre le demandeur et le service. Un bus de service d'entreprise permet aux fonctions métier de d'échanger des informations dans des formats différents, le bus garantissant que l'information distribuée à une fonction métier a le format correspondant à l'application.
- *Traitement* des événements métier provenant de sources différentes. Un bus de service d'entreprise prend en charge les interactions basées sur l'événement en plus des échanges de message pour traiter les demandes de service.

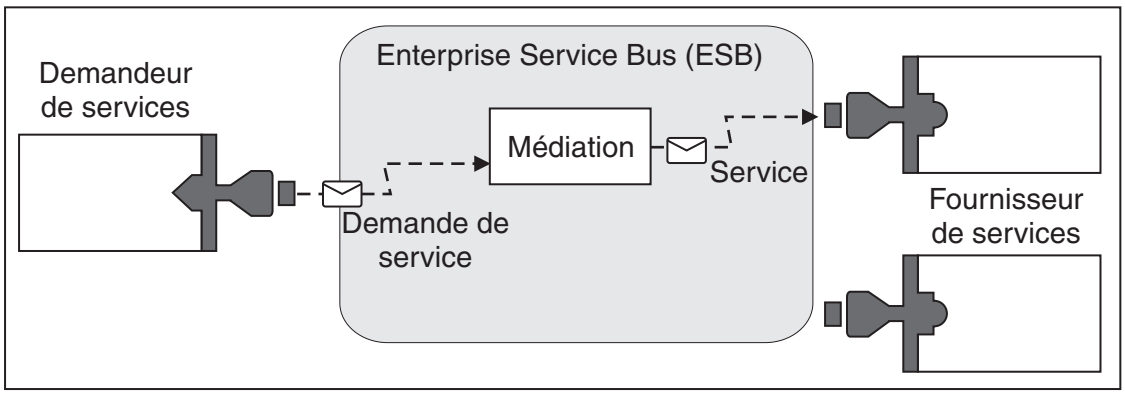


Figure 42. Bus de service d'entreprise. Le bus de service d'entreprise achemine les messages entre les applications, qui sont demandeurs ou fournisseurs de services. Le bus convertit les protocoles de transport ainsi que les formats des messages entre les demandeurs et les fournisseurs. Dans ce schéma, chaque application utilise un protocole différent (représenté par les différentes formes géométriques de leurs connecteurs) et utilise différents formats de message.

En utilisant le bus de service d'entreprise, vous vous consacrez désormais entièrement à votre coeur de métier, sans vous soucier des systèmes informatiques. Vous pouvez apporter des modifications ou des ajouts aux services quand vous avez besoin, par exemple, pour répondre aux évolutions de vos besoins métier, augmenter les capacités de service ou ajouter de nouvelles fonctionnalités. Vous pouvez effectuer vos modifications en redéfinissant le bus, avec très peu ou pas d'incidence sur les services et les applications existantes qui utilisent le bus.

Infrastructure de messagerie Enterprise Service Bus

IBM Business Process Manager inclut des fonctions de bus de services d'entreprise. IBM Business Process Manager prend en charge l'intégration de technologies orientées services, orientées messages et gérées par des événements pour fournir une infrastructure de messagerie normalisée au sein d'un bus de services d'entreprise intégré.

Les fonctions de services d'entreprise qui vous pouvez utiliser pour les applications d'entreprise fournissent non seulement une couche de transport mais également un support de médiation pour faciliter les interactions des services. Le bus de services d'entreprise s'appuie sur des normes ouvertes et l'architecture SOA (Service Oriented Architecture). Il repose sur l'infrastructure robuste Java EE et les services de plateformes associés fournis par IBM WebSphere Application Server Network Deployment.

IBM Business Process Manager est doté de la même technologie qu'IBM WebSphere Enterprise Service Bus. Cette technologie fait partie des fonctionnalités sous-jacentes de IBM Business Process Manager et aucune licence supplémentaire de WebSphere Enterprise Service n'est nécessaire pour en tirer parti.

Toutefois, vous pouvez déployer des licences autonomes supplémentaires de WebSphere Enterprise Service Bus dans l'entreprise pour étendre la connectivité des solutions d'intégration de processus IBM Business Process Manager. Par exemple, WebSphere Enterprise Service Bus peut être installé à proximité d'une application SAP pour héberger IBM WebSphere Adapter for SAP et transformer des messages avant d'envoyer des informations sur le réseau à un processus métier organisé par IBM Business Process Manager.

Hôtes de messagerie ou de destination de file d'attente :

Un hôte de messagerie ou de destination de file d'attente constitue la fonction de messagerie au sein d'un serveur. Un serveur devient l'hôte de destination des messages lorsque vous le configurez en tant que cible de messagerie.

Le moteur de messagerie s'exécute dans le serveur. Le moteur de messagerie assure des fonctions de messagerie et constitue un point de connexion entre les applications et le bus. La communication asynchrone de l'architecture SCA (Service Component Architecture), les importations et les exportations JMS et le traitement interne asynchrone utilisent les files d'attente de messages sur le moteur de messagerie.

L'environnement de déploiement connecte la source de messages à la cible des messages via le bus, lorsque les modules d'applications sont déployés. Si vous connaissez la source et la cible des messages, vous pouvez déterminer plus facilement le type d'environnement de déploiement dont vous avez besoin.

Les données rémanentes peuvent être stockées dans un magasin de données par les applications. Un magasin de données est un ensemble de tables contenu dans une base de données ou dans un schéma, ou encore dans un magasin de données. Le moteur de messagerie utilise une instance d'une source de données JDBC pour interagir avec cette base de données.

Configurez l'hôte de destination des messages lorsque vous définissez votre environnement de déploiement **Server** à partir de la console d'administration, ou désignez le serveur en tant qu'hôte cible durant l'installation de logiciels.

Magasins de données :

Chaque moteur de messagerie peut utiliser un magasin de données (un magasin de données constitue un ensemble de tables contenues dans une base de données ou dans un schéma incluant des données permanentes).

Toutes les tables du magasin sont contenues dans le même schéma de base de données. Vous pouvez créer chaque magasin de données dans une base de données séparée. Il est également possible de créer plusieurs magasins dans la même base, chaque magasin utilisant un schéma différent.

Un moteur de messagerie utilise une instance d'une source de données JDBC pour interagir avec la base de données qui contient le magasin de données pour ce moteur de messagerie.

Fournisseurs JDBC :

Vous pouvez utiliser les fournisseurs JDBC pour que les applications puissent interagir avec les bases de données relationnelles.

Les applications utilisent des fournisseurs JDBC pour interagir avec des bases de données relationnelles. Le fournisseur JDBC fournit la classe d'implémentation de pilote JDBC qui est nécessaire pour accéder à

une base de données spécifique. Pour créer un pool de connexions à cette base de données, associez une source de données au fournisseur JDBC. Ensemble, le fournisseur JDBC et les objets de source de données présentent les mêmes fonctions que la fabrique de connexions Java Connector Architecture (JCA), qui assure la connexion à une base de données non relationnelle.

Voir les exemples de configuration d'environnement autonome type et de configuration d'environnement de déploiement type dans la rubrique précédente.

Pour plus d'informations sur les fournisseurs JDBC, voir «Fournisseurs JDBC» dans le centre de documentation de WebSphere Application Server.

Bus d'intégration de services pour IBM Business Process Manager :

Un bus d'intégration de services est un mécanisme de communications géré prenant en charge l'intégration de services via une messagerie synchrone et asynchrone. Un bus se compose de moteurs de messagerie interconnectés gérant les ressources de bus. Il représente l'une des technologies WebSphere Application Server sur lesquelles repose IBM Business Process Manager.

Un bus d'intégration de services unique et moteur de messagerie unique utilise le même schéma de base de données que la base de données produit par défaut. Chaque environnement de déploiement possède son propre bus. Le bus unique est appelé **BPM.*nom_environnement_déploiement*.Bus**.

Une destination de bus est une adresse logique vers laquelle les applications peuvent définir une liaison en tant que fournisseur, consommateur, ou les deux. Une destination de file d'attente est une destination de bus utilisée pour la messagerie point-à-point.

Chaque bus peut contenir un ou plusieurs membres, chacun d'eux étant soit un serveur, soit un cluster.

Le terme de *topologie de bus* se rapporte à l'organisation physique entre les serveurs d'applications, les moteurs de messagerie et les gestionnaires de files d'attente WebSphere MQ, ainsi que le modèle de connexions et de liaisons de bus intermédiaires qui composent le bus ESB.

Modules et applications de service

Un module de service est un module SCA (Service Component Architecture) qui offre des services dans l'environnement d'exécution. Lorsque vous déployez un module de service sur IBM Business Process Manager, vous générez l'application de service associée conditionnée sous la forme d'un fichier EAR (enterprise archive).

Les modules de service sont les unités de base d'un déploiement et peuvent contenir des composants, des bibliothèques et des modules de transfert dont se sert l'application de service associée. Les modules de service disposent d'exportation et à titre facultatif d'importations pour définir les relations entre modules et demandeurs et fournisseurs de services. WebSphere Process Server prend en charge les modules pour les services métier et les modules de médiation. Les modules et les modules de médiation constituent des types de modules SCA. Un module de communication permet les communications entre applications en transformant l'appel de service dans un format compris par la cible, en transmettant la demande à la cible et en renvoyant le résultat au module émetteur. Un module pour un service métier met en oeuvre la logique d'un processus métier. Toutefois, un module peut aussi inclure la même logique de médiation que celle conditionnée dans le module de médiation.

Déploiement d'une application de service

Le processus de déploiement d'un fichier EAR contenant une application de service est identique à celui de tout fichier EAR. Vous pouvez modifier les valeurs des paramètres de médiation lors de la phase d'exécution. Après avoir déployé un fichier EAR contenant un module SCA, vous pouvez visualiser les informations sur l'application de service et son module associé. Vous pouvez visualiser la manière dont

un module de service est connecté au demandeurs de services (via les exportations) et aux fournisseurs de services (via les importations).

Affichage des détails d'un module SCA

Les informations de module de service que vous pouvez afficher dépendent du module SCA. Elles comprennent les attributs suivants.

- Nom du module SCA
- Description du module SCA
- Nom de l'application associée
- Version du module SCA, si le module est versionné
- Importations du module SCA :
  - Les interfaces d'importation sont des définitions abstraites qui décrivent la façon dont un module SCA accède à un service.
  - Les liaisons d'importation sont des définitions concrètes qui indiquent le mécanisme physique via lequel un module SCA accède à un service. Par exemple, via SOAP/HTTP.
- Exportations du module SCA :
  - Les interfaces d'exportation sont des définitions abstraites qui décrivent la façon dont les demandeurs de services accèdent à un module SCA.
  - Les liaisons d'exportation sont des définitions concrètes qui indiquent le mécanisme physique via lequel un demandeur de services accède à un modules SCA et indirectement un service.
- Propriétés de module SCA

Importations et liaisons d'importation :

Les importations définissent les interactions entre les modules SCA et les fournisseurs de services. Les modules SCA utilisent des importations pour permettre aux composants d'accéder aux services externes (services qui se trouvent en dehors du module SVA) à l'aide d'une représentation locale. Les liaisons d'importation définissent la façon spécifique dont on accède à un service externe.

Si les modules SCA n'ont pas besoin d'accéder à des services externes, ils n'ont pas besoin de disposer d'importations. Les modules de médiation disposent généralement d'une ou de plusieurs importations qui sont utilisées pour transmettre les messages ou demandes sur leurs cibles prévues.

Interfaces et liaisons

Une importation de modules SCA requiert au moins une interface et qu'une importation de modules SCAN ait une seule liaison.

- Les interfaces d'exportation sont des définitions abstraites qui définissent un ensemble d'opérations à l'aide de WSDL (Web Services Description Language), un langage XML permettant de décrire des services Web. Un module SCA peut avoir un grand nombre d'interfaces.
- Les liaisons d'importation sont des définitions concrètes qui spécifient le mécanisme physique utilisé par les modules SCA pour accéder à un service externe.

Liaisons d'importation prises en charge

IBM Business Process Manager prend en charge les liaisons d'importation ci-après :

- Les liaisons SCA connectent les modules SCA à d'autres modules SCA. Les liaisons SCA sont également appelées liaisons par défaut.
- Les liaisons de service Web permettent aux composants d'appeler des services Web. Les protocoles pris en charge sont SOAP1.1/HTTP, SOAP1.2/HTTP et SOAP1.1/JMS.

Vous pouvez utiliser une liaison SOAP1.1/HTTP ou SOAP1.2/HTTP basée sur JAX-WS (Java API for XML Web Services), qui permet l'interaction avec les services à l'aide de liaisons de document ou de liaisons littérales RPC et qui utilise des gestionnaires JAX-WS pour personnaliser les appels. Une liaison SOAP1.1/HTTP distincte est fournie pour permettre l'interaction avec les services qui utilisent une liaison codée RPC ou lorsque des gestionnaires JAX-RPC doivent être utilisés pour personnaliser les appels.

- Les liaisons HTTP permettent d'accéder aux applications à l'aide du protocole HTTP.
- Les liaisons d'importation EJB (Enterprise JavaBeans)permettent aux composants SCA d'appeler des services fournis par la logique métier Java EE exécutée sur un serveur Java EE.
- Les liaisons de système d'information d'entreprise (EIS) permettent la connectivité entre les composants SCA et un système EIS externe. Cette communication est obtenue en utilisant des adaptateurs de ressources.
- Les liaisons Java Message Service (JMS) 1.1 permettent l'interopérabilité avec le fournisseur de messagerie par défaut de WebSphere Application Server. JMS peut exploiter divers types de protocoles de transport, tels que TCP/IP et HTTPS. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge.
- Les liaisons JMS génériques permettent l'interopérabilité avec les fournisseurs JMS tiers qui s'intègrent à WebSphere Application Server à l'aide de la fonction JMS Application Server Facility (ASF).
- Les liaisons JMS WebSphere MQ permettent l'interopérabilité avec les fournisseurs JMS basés sur WebSphere MQ. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge. Si vous souhaitez utiliser WebSphere MQ comme fournisseur JMS, utilisez des liaisons JMS WebSphere MQ.
- Les liaisons WebSphere MQ permettent l'interopérabilité avec WebSphere MQ. Vous ne pouvez utiliser des liaisons WebSphere MQ qu'avec des gestionnaires de file d'attente distants via une connexion client WebSphere MQ ; en effet, vous ne pouvez pas les utiliser avec des gestionnaires de file d'attente locaux. Utilisez des liaisons WebSphere MQ si vous souhaitez communiquer avec des applications natives WebSphere MQ.

Appel dynamique de services

Les services peuvent être appelés via n'importe quelle liaison d'importation prise en charge. Un service se trouve généralement sur un point de contact spécifié lors de l'importation. Ce point de contact est appelé point de contact statique. Il est possible d'appeler un service différent en remplaçant le point de contact statique. Le remplacement dynamique des points de contact statiques vous permet d'appeler un service sur un autre point de contact, via n'importe quelle liaison d'importation prise en charge. L'appel dynamique de services vous permet également d'appeler un service pour lequel la liaison d'importation prise en charge n'a pas de point de contact statique.

Une importation avec une liaison associée est utilisée pour spécifier le protocole et sa configuration pour l'appel dynamique. L'importation utilisée pour l'appel dynamique peut être reliée au composant appelant ou être sélectionnée dynamiquement lors de la phase d'exécution.

Pour les appels de service Web et SCA, il est également possible d'effectuer un appel dynamique sans importation, avec le protocole et la configuration extraite de l'URL du point de contact. Le type de cible de l'appel est identifié à partir de l'adresse URL du point de contact. Si une importation est utilisée, l'URL doit être compatible avec le protocole de la liaison d'importation.

- Une adresse URL SCA URL indique l'appel d'un autre module SCA.
- Une adresse URL HTTP ou JMS par défaut indique l'appel d'un service Web ; pour ces adresses URL, il est possible de fournir une valeur supplémentaire pour le type de liaison qui indique que l'adresse URL représente un appel via une liaison HTTP ou JMS.
- Pour une adresse URL HTTP de service Web, le protocole SOAP 1.1 est utilisé par défaut et une valeur pour le type de liaison indiquant l'utilisation de SOAP 1.2 peut être spécifiée.

Exportation et liaisons d'exportation :

Les exportations définissent les interactions entre les modules SCA et les demandeurs de services. Les modules SCA utilisent des exportations pour proposer des services aux autres. Les liaisons d'exportation définissent le mode d'accès à un module SCA par les demandeurs de service.

Interfaces et liaisons

Une exportation de module SCA nécessite au moins une interface.

- Les interfaces d'exportation sont des définitions abstraites qui définissent un ensemble d'opérations à l'aide de WSDL (Web Services Description Language), un langage XML permettant de décrire des services Web. Un module SCA peut disposer d'un grand nombre d'interfaces d'exportation.
- Les liaisons d'exportation sont des définitions concrètes qui spécifient le mécanisme physique utilisé par les demandeurs de services pour accéder à un service. En règle générale, une exportation de module SCA a une liaison définie. Une exportation, pour laquelle aucune liaison n'a été spécifiée, est interprétée par l'environnement d'exportation comme une exportation dotée d'une liaison de type SCA.

Liaisons d'exportation prises en charge

Le IBM Business Process Manager prend en charge les liaisons d'exportation ci-après :

- Les liaisons SCA connectent les modules SCA à d'autres modules SCA. Les liaisons SCA sont également appelées liaisons par défaut.
- Les liaisons de services Web permettent d'appeler des exportations en tant que services Web. Les protocoles pris en charge sont SOAP1.1/HTTP, SOAP1.2/HTTP et SOAP1.1/JMS.

  Vous pouvez utiliser une liaison SOAP1.1/HTTP ou SOAP1.2/HTTP basée sur JAX-WS (Java API for XML Web Services), qui permet l'interaction avec les services à l'aide de liaisons de document ou de liaisons littérales RPC et qui utilise des gestionnaires JAX-WS pour personnaliser les appels. Une liaison SOAP1.1/HTTP distincte est fournie pour permettre l'interaction avec les services qui utilisent une liaison codée RPC ou lorsque des gestionnaires JAX-RPC doivent être utilisés pour personnaliser les appels.
- Les liaisons HTTP permettent d'accéder aux exportations à l'aide du protocole HTTP.
- Les liaisons d'exportation EJB (Enterprise JavaBeans) permettent d'exposer les composants SCA comme des EJB pour que la logique métier de Java EE puisse appeler les composants SCA qui ne seraient autrement pas disponibles.
- Les liaisons de système d'information d'entreprise (EIS) permettent la connectivité entre les composants SCA et un système EIS externe. Cette communication est obtenue en utilisant des adaptateurs de ressources.
- Les liaisons Java Message Service (JMS) 1.1 permettent l'interopérabilité avec le fournisseur de messagerie par défaut de WebSphere Application Server. JMS peut exploiter divers types de protocoles de transport, tels que TCP/IP et HTTPS. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge.
- Les liaisons JMS génériques permettent l'interopérabilité avec les fournisseurs JMS tiers qui s'intègrent à WebSphere Application Server à l'aide de la fonction JMS Application Server Facility (ASF).
- Les liaisons JMS WebSphere MQ permettent l'interopérabilité avec les fournisseurs JMS basés sur WebSphere MQ. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge. Si vous souhaitez utiliser WebSphere MQ comme fournisseur JMS, utilisez des liaisons JMS WebSphere MQ.
- Les liaisons WebSphere MQ permettent l'interopérabilité avec WebSphere MQ. Utilisez une connexion éloignée (ou client) pour vous connecter à un gestionnaire de files d'attente MQ sur un poste distant. Une connexion (ou liaisons) locale correspond à une connexion directe à WebSphere MQ. Elles ne peuvent être utilisées que pour la connexion à un gestionnaire de files d'attente MQ sur une même machine. WebSphere MQ autorise les deux types de connexion, mais les liaisons MQ prennent en charge uniquement la connexion "à distance" (ou "client") .

Modules de médiation :

Les modules de médiation sont des modules SCA qui peuvent modifier le format, le contenu ou la cible des demandes de services.

Les modules de médiation s'appliquent aux messages circulant entre les demandeurs et les fournisseurs de services. Vous pouvez acheminer des messages vers différents fournisseurs de services et de modifier le format ou le contenu du message. Les modules de médiation peuvent fournir des fonctions telles que la consignation des messages et le traitement des erreurs, personnalisées en fonction de votre environnement.

Vous pouvez modifier certains aspects des modules de médiation à partir de la console d'administration , sans que le redéploiement du module soit nécessaire.

Composants des modules de médiation

Les modules de médiation contiennent les éléments suivants :

- Importations qui définissent des interactions entre les modules SCA et les fournisseurs de services. Elles permettent aux modules SCA d'appeler des services externes comme s'ils étaient locaux. Vous pouvez visualiser les importations du module de médiation et modifier la liaison.
- Exportations qui définissent des interactions entre les modules SCA et les demandeurs de services. Elles permettent à un module SCA d'offrir un service et de définir les interfaces externes (points d'accès) d'un module SCA. Vous pouvez visualiser des exportations de module de médiation.
- Composants SCA, blocs structurels des modules SCA tels que les modules de médiation. Vous pouvez créer et personnaliser des composants et des modules SCA graphiquement via Integration Designer. Après avoir déployé un module de médiation, vous pouvez en personnaliser certains aspects à partir de la console d'administration , sans que le redéploiement du module soit nécessaire.

  Généralement, les modules de médiation contiennent un type spécifique de composant SCA appelé *composant de flux de médiation*. Les composants de flux de médiation permettent de définir ces flux.

  Un composant de flux de médiation peut contenir aucune, une ou plusieurs primitives de médiation. IBM Business Process Manager prend en charge un ensemble fourni de primitives de médiation qui fournissent des fonctionnalités pour l'acheminement et la transformation de messages. Si vous avez besoin d'une primitive de médiation plus souple, utilisez la primitive de médiation personnalisée pour appeler la logique personnalisée.

  L'objet d'un module de médiation qui ne contient pas de composant de flux de médiation est de convertir des demandes de services d'un protocole à un autre. Par exemple, une demande de service peut être effectuée via SOAP/JMS, mais risque d'avoir besoin d'être transformée en SOAP/HTTP avant d'être envoyée.

  Remarque : Vous pouvez afficher les modules de médiation et leur apporter certaines modifications à partir d'IBM Business Process Manager. Cependant, il n'est pas possible de visualiser ni de modifier des composants SCA à l'intérieur d'un module à partir d'IBM Business Process Manager. Utilisez Integration Designer pour personnaliser les composants SCA.

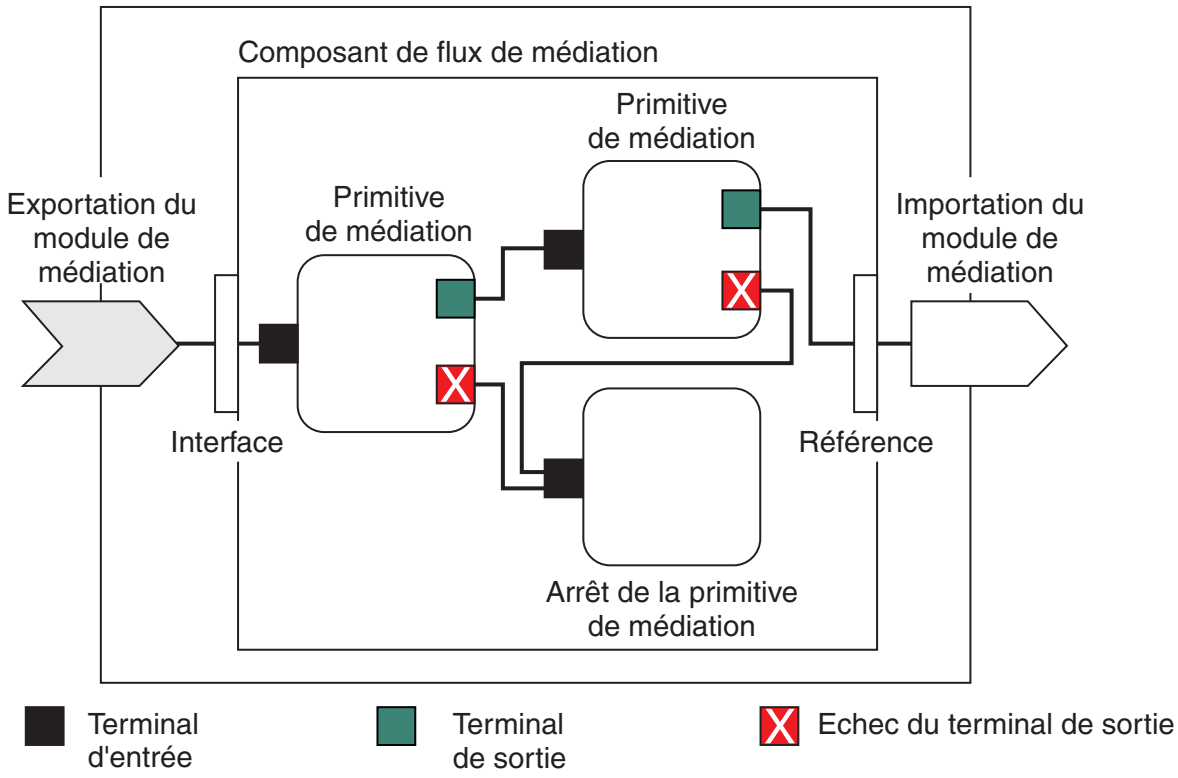


Figure 43. Exemple simplifié d'un module de médiation. Le module de médiation contient un composant de flux de médiation, qui contient des primitives de médiation.

- Propriétés

  Les propriétés de certaines primitives de médiation peuvent être affichées sur la console d'administration en tant que propriétés complémentaires d'un module SCA.

  Pour que les propriétés de la primitive de médiation soient visibles à partir de la console d'administration d'IBM Business Process Manager, le développeur d'intégration doit promouvoir les propriétés. Certaines propriétés peuvent être configurées administrativement et Integration Designer les décrit comme étant des propriétés pouvant être promues du cycle d'intégration au cycle d'administration. La raison pour laquelle d'autres propriétés sont incompatibles avec une configuration administrative est que leur modification affecte le flux de médiation d'une manière qui nécessite le redéploiement du module de médiation. Integration Designer répertorie les propriétés que vous pouvez promouvoir dans la liste des Propriétés promues d'une primitive de médiation.

  Vous pouvez utiliser la console d'administration d'IBM Business Process Manager pour modifier la valeur des propriétés promues sans qu'il soit nécessaire de redéployer un module de médiation ni de redémarrer le serveur ou le module.

  Généralement, les flux de médiation utilisent immédiatement les modifications de propriété. Toutefois, si les modifications de propriété se produisent dans une cellule de gestionnaire de déploiement, elles prendront effet sur chaque noeud à chaque fois qu'il sera synchronisé. Par ailleurs, les flux de médiation qui sont en transit continuent d'utiliser les valeurs précédentes.

  Remarque : A partir de la console d'administration, vous ne pouvez modifier que les valeurs de propriété et non pas les groupes, noms ou types de propriété. Si vous souhaitez modifier les groupes, noms ou types de propriété, vous devez utiliser Integration Designer.
- Un module de médiation ou une bibliothèque dépendante peut également définir des sous-flux. Un sous-flux encapsule un ensemble de primitives de médiation connectées les unes aux autres comme un élément réutilisable d'une logique d'intégration. Une primitive peut être ajoutée à un flux de médiation pour appeler un sous-flux.

Déploiement de modules de médiation

Les modules de médiation sont créés via Integration Designer et sont généralement déployés sur IBM Business Process Manager dans un fichier EAR (fichier d'archive d'entreprise).

La valeur des propriétés promues peut être modifiée lors du déploiement.

Vous pouvez exporter un module de médiation à partir de Integration Designer, puis ordonner à Integration Designer de compiler le module de médiation dans un fichier JAR (archive Java), lequel est ensuite intégré à un fichier EAR. Vous pouvez maintenant déployer le fichier EAR en installant une nouvelle application à partir de la console d'administration.

Les modules de médiation peuvent être considérés comme une entité. En revanche, les modules SCA sont définis par un certain nombre de fichiers XML stockés dans un fichier JAR.

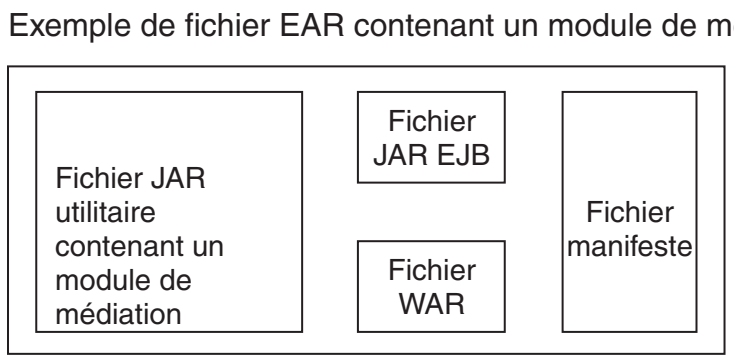


Figure 44. Exemple simplifié de fichier EAR contenant un module de médiation. Le fichier EAR contient des fichiers JAR. Le fichier JAR utilitaire contient un module de médiation.

Primitives de médiation :

Les composants de flux de médiation agissent sur les flux de messages entre les composants de service. Les fonctionnalités d'un composant de médiation sont implémentées par les *primitives de médiation*, qui mettent en oeuvre des types d'implémentation de service standard.

Un composant de flux de médiation dispose d'un ou plusieurs flux. Par exemple, un pour la requête et un pour la réponse.

IBM Business Process Manager prend en charge un ensemble de primitives de médiation qui implémentent des fonctionnalités de médiation standard pour les modules de médiation ou modules déployés vers IBM Business Process Manager. Si vous avez besoin d'utiliser des fonctions de médiation spéciales, vous pouvez développer vos propres primitives de médiation.

Une primitive de médiation définit une opération «entrante» qui traite ou gère les messages représentés sous forme d'objets SMO (Service Message Object). Elle peut également définir une opération «sortante» qui envoie des messages vers un autre composant ou module.

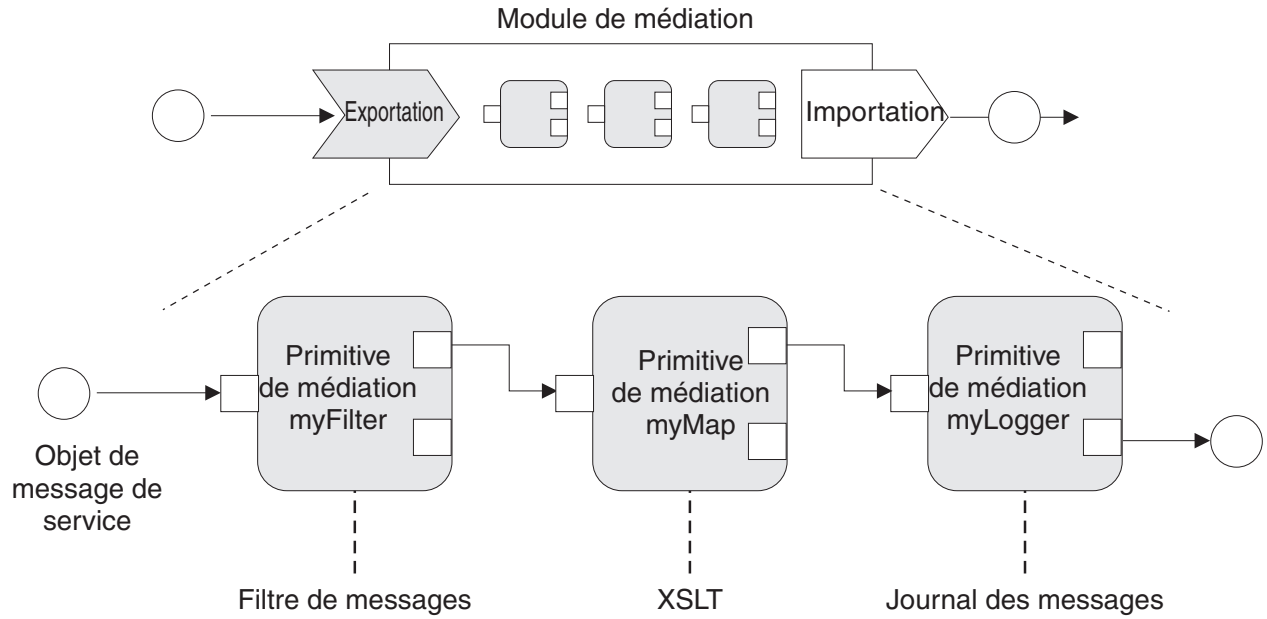


Figure 45. Module de médiation contenant trois primitives de médiation

Vous pouvez utiliser Integration Designer pour configurer les primitives de médiation et définir leurs propriétés. Certaines de ces propriétés peuvent être visibles pour l'administrateur d'exécution si elles ont été promues. Toute propriété primitive de médiation qui peut être promue peut également être une propriété dynamique. Une propriété dynamique peut être remplacée, en phase d'exécution, à l'aide d'un fichier de règles.

Integration Designer permet également de modéliser et d'assembler sous forme graphique les composants de flux de médiation à partir de primitives de médiation et d'assembler les modules de médiation ou les modules depuis des composants de flux de médiation. La console d'administration fait référence aux modules de médiation et modules en tant que modules SCA.

Integration Designer permet également la définition de sous-flux dans les modules ou leurs bibliothèques dépendantes. Un sous-flux peut contenir toute primitive de médiation excepté pour la primitive de médiation Résolution de règle. Un sous-flux est appelé à partir d'un flux de demandes ou de réponses ou à partir d'un autre sous-flux utilisant la primitive de médiation Sous-flux. Les propriétés promues à partir des primitives de médiation dans un sous-flux sont affichées en tant que propriétés sur les primitives de médiation Sous-flux. Elles peuvent ensuite être à nouveau promues jusqu'à ce qu'elles atteignent le niveau de module auquel elles peuvent être modifiées par l'administrateur d'exécution.

Primitives de médiation prises en charge

L'ensemble suivant de primitives de médiation est pris en charge par IBM Business Process Manager:

Mappe d'objet métier
Transforme les messages.

- Définit les transformations de message à l'aide d'une mappe d'objet métier, qui peut être réutilisée.
- Permet de définir les transformations de message sous forme graphique, à l'aide de l'éditeur de mappe d'objet métier.
- Peut modifier le contenu d'un message.
- Peut transformer un type de message d'entrée en un type de message de sortie différent.

Médiation personnalisée

Permet d'implémenter votre propre logique de médiation en code Java. La primitive de médiation personnalisée associe la flexibilité d'une primitive de médiation définie par l'utilisateur, à la simplicité d'une primitive de médiation prédéfinie. Vous pouvez créer des patterns d'acheminement et de transformation complexes en :

- Créant le code Java.
- Créant vos propres propriétés.
- Ajoutant de nouveaux terminaux.

Vous pouvez appeler un service depuis une primitive de médiation personnalisée, mais la primitive de médiation d'appel de service (Service Invoke) est conçue pour appeler des services et fournir d'autres fonctions, notamment de relance.

Gestionnaire de données

Vous permet de transformer une partie d'un message. Il est utilisé pour convertir un élément de message d'un format physique en une structure logique ou d'une structure logique en format physique. L'utilisation principale de la primitive consiste à convertir un format physique, comme une chaîne de texte au sein d'un objet de message texte JMS, en une structure d'objet métier logique et inversement. Cette médiation est généralement utilisée pour :

- Transformer une section du message entrant d'une structure définie en une autre, par exemple lorsque l'objet SMO comprend une valeur de chaîne délimitée par une virgule et que vous voulez faire une analyse syntaxique dans un objet métier spécifique.
- Modifier le type de message – par exemple lorsqu'une exportation JMS a été configurée pour utiliser une liaison de données de type de base JMS et qu'au sein du module de médiation, le développeur d'intégration décide que le contenu doit être gonflé en une structure BO.

Consultation de base de données

Modifie les messages, à l'aide d'informations provenant d'une base de données fournie par l'utilisateur.

- Vous devez définir une base de données, une source de données et tous les paramètres d'authentification du serveur que doit utiliser la primitive de médiation Recherche de base de données à utiliser. Utilisez la console d'administration pour vous simplifier la tâche.
- La primitive de médiation Recherche de base de données ne peut lire qu'à partir d'une seule table.
- La colonne de clé spécifiée doit contenir une valeur unique.
- Les données des colonnes de valeur doivent être d'un type de schéma XML simple ou d'un type de schéma XML permettant d'étendre un type de schéma XML simple.

Consultation de point de contact (Endpoint Lookup)

Permet d'effectuer le routage dynamique de requêtes en recherchant les points de contact de services dans un référentiel.

- Les informations relatives au point de contact de service sont extraites de WebSphere Service Registry and Repository (WSRR). Le registre WSRR peut être local ou distant.
- Vous effectuez les modifications du registre à partir de la console d'administration WSRR.
- IBM Business Process Manager doit connaître le registre à utiliser, par conséquent, vous devez créer des définitions d'accès WSRR à l'aide de la console d'administration IBM Business Process Manager.

Emetteur d'événements

Améliore le contrôle en vous laissant envoyer des événements à partir d'un composant de flux de médiation.

- Vous pouvez suspendre l'action de médiation en décochant la case.
- Vous pouvez visualiser ces événements via le navigateur CBE (Common Base Event) de IBM Business Process Manager.

- Vous pouvez uniquement envoyer des événements vers un point significatif d'un flux de médiation, à des fins de performances.
- Vous pouvez définir les parties du message que contient l'événement.
- Les événements sont envoyés suivant le format Common Base Events (CBE) vers un serveur Common Event Infrastructure (CEI).
- Pour pouvoir exploiter pleinement les informations sur les émetteurs d'événements, les consommateurs d'événements doivent comprendre parfaitement la structure Common Base Events. Le format Common Base Events est caractérisé par un schéma global, mais qui ne modélise pas les données spécifiques à l'application contenues dans les éléments de données étendus. Afin de modéliser les éléments de données étendus, les outils Integration Designer génèrent un fichier de définitions pour le catalogue d'événements Common Event Infrastructure (CEI), pour chacune des primitives de médiation Emetteur d'événement. Les fichiers de définitions du catalogue d'événements sont des artefacts d'exportation destinés à vous venir en aide ; ils ne sont pas utilisés par Integration Designer ou par le programme d'exécution IBM Business Process Manager. Il convient de vous référer aux fichiers de définitions du catalogue d'événements lorsque vous créez des applications destinées à consommer des événements générés par un émetteur d'événements.
- Vous pouvez spécifier d'autres options de contrôle à partir de IBM Business Process Manager. Ainsi, vous pouvez surveiller les événements émis à partir d'importations et d'exportations.

Fail (Echec)
Arrête un chemin donné dans le flux, et génère une exception.

Fan In (Entrance)
Permet de regrouper (d'associer) des messages.

- Est utilisable uniquement en association avec la primitive de médiation de sortance.
- L'association des primitives de médiation d'entrance et de sortance permet le regroupement de données dans un message de sortie.
- La primitive de médiation d'entrance reçoit des messages jusqu'à ce qu'un point de décision soit atteint, puis un message est sorti.
- Le contexte partagé permet de conserver les données de regroupement.

Fan Out (Sortance)
Permet de diviser et de regrouper (associer) des messages.

- L'association des primitives de médiation d'entrance et de sortance permet le regroupement de données dans un message de sortie.
- En mode d'itération, la primitive de médiation de sortance vous permet d'itérer via un seul message d'entrée contenant un élément répétitif. Pour chaque occurrence de l'élément répétitif, un message est envoyé.
- Le contexte partagé permet de conserver les données de regroupement.

Configurateur d'en-tête HTTP
Fournit un mécanisme de gestion des en-têtes dans les messages HTTP.

- Peut créer, définir, copier ou supprimer des en-têtes de messages HTTP.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes HTTP.

Mappage
Transforme les messages.

- Permet d'effectuer des transformations XSL (Extensible Stylesheet Language) ou des transformations de mappe d'objets métier.
- Vous transformez les messages avec la transformation XSLT 1.0, XSLT 2.0 ou de mappe d'objets métier. Les transformations XSL opèrent sur une sérialisation XML du message, tandis que les transformations de Mappe d'objet métier opèrent sur les objets SDO.

Configurateur d'élément de message

Fournit un système simple permettant de définir le contenu des messages.

- Permet de modifier, ajouter ou supprimer les éléments du message.
- Ne modifie pas le type du message.
- Les données des colonnes de valeur doivent être d'un type de schéma XML simple ou d'un type de schéma XML permettant d'étendre un type de schéma XML simple.

Filtre de message

Achemine les messages par différent chemins, selon le contenu du message.

- Vous pouvez suspendre l'action de médiation en décochant la case.

Journal des messages

Consigne les messages dans une base de données relationnelle ou via votre propre consignateur personnalisé. Les messages sont stockés au format XML, c'est pourquoi les données peuvent subir un post-traitement par des applications compatibles XML.

- Vous pouvez suspendre l'action de médiation en décochant la case.
- Le schéma de base de données rationnel (structure de table) est défini par IBM.
- Par défaut, la primitive de médiation Message Logger utilise la base de données Common. L'environnement d'exécution mappe la source de données de **jdbc/mediation/messageLog** dans la base de données Common.
- Vous pouvez définir les classes d'implémentation Handler pour personnaliser le comportement du consignateur personnalisé. Vous pouvez éventuellement fournir des classes d'implémentation Formatter, Filter ou les deux pour personnaliser le comportement du consignateur personnalisé.

Configurateur d'en-tête MQ

Fournit un mécanisme de gestion des en-têtes dans les messages MQ.

- Peut créer, définir, copier ou supprimer des en-têtes de messages MQ.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes MQ.

Résolution de règle

Permet la configuration dynamique des demandes en recherchant les points de contact de services et les fichiers de règles associés dans un référentiel.

- Vous pouvez utiliser un fichier de règles pour remplacer de manière dynamique les propriétés promues d'autres primitives de médiation.
- Les informations relatives au point de contact de service et les informations de règle sont extraites de WebSphere Service Registry and Repository (WSRR). Le registre WSRR peut être local ou distant.
- Vous effectuez les modifications du registre à partir de la console d'administration WSRR.
- IBM Business Process Manager doit connaître le registre à utiliser, par conséquent, vous devez créer des définitions d'accès WSRR à l'aide de la console d'administration IBM Business Process Manager.

Invocation de service

Appel un service depuis un flux de médiation, au lieu d'attendre jusqu'à la fin du flux de médiation et d'utiliser un système d'appel.

- Si le service renvoie une erreur, vous pouvez tenter de nouveau le même service ou appeler un autre service.
- La primitive de médiation d'appel de service (Service Invoke) est une primitive de médiation puissante qui peut être utilisée pour des appels de service simples, ou conjointement avec d'autres primitives de médiation pour les médiations complexes.

Définir un type de message

En phase de développement d'intégration, permet de traiter des zones de messages faiblement

typées comme s'il s'agissait de zones fortement typées. Une zone est faiblement typée si elle peut contenir plusieurs type de données. Une zone est fortement typée si son type et sa structure interne sont connus.

- En phase d'exécution, la primitive de médiation Définition du type de message vous permet de vérifier que le contenu d'un message correspond aux types de données attendus.

Configurateur d'en-tête SOAP
Fournit un mécanisme de gestion des en-têtes dans les messages SOAP.

- Peut créer, définir, copier ou supprimer des en-têtes de messages SOAP.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes SOAP.

Arrêter
Arrête un chemin donné dans le flux sans générer d'exception.

Filtre de type
Permet d'acheminer des messages vers un chemin différent d'un flux, en fonction de leur type.

WebSphere eXtreme Scale Extraire
Vous pouvez extraire les informations depuis un environnement de cache de serveur eXtreme Scale.

- Vous pouvez rechercher des valeurs dans le cache, et les stocker en tant qu'éléments du message à l'aide d'une clé.
- En combinant les primitives de médiation eXtreme Scale Stocker et Extraire, vous pouvez mettre en cache la réponse à partir d'un système dorsal. Les requêtes futures n'auront pas besoin d'accéder à ce système dorsal.
- Vous devez créer des définitions de eXtreme Scale à l'aide de la console d'administration WebSphere ESB, afin de pouvoir indiquer quel serveur eXtreme Scale utiliser.

WebSphere eXtreme Scale Stocker
Vous pouvez stocker les informations dans un environnement de cache de serveur eXtreme Scale.

- Vous pouvez stocker des informations dans un cache eXtreme Scale à l'aide d'une clé et d'un objet.
- En combinant des primitives de médiation deeXtreme Scale Stocker et Extraire, vous pouvez utiliser la primitive de médiation Stocker pour stocker des données dans le cache, et utiliser la primitive de médiation Extraire pour extraire les données précédemment stockées dans le cache.
- Vous devez créer des définitions de eXtreme Scale à l'aide de la console d'administration WebSphere ESB, afin de pouvoir indiquer quel serveur eXtreme Scale utiliser.

Routage dynamique :

Vous pouvez acheminer les messages de plusieurs manière à l'aide de points de contact définis en phase d'intégration ou de points de contact déterminés, de manière dynamique, en phase d'exécution.

Le routage dynamique comprend deux cas de routage des messages :

- Le routage dynamique dans lequel le flux est dynamique, mais avec tous les noeuds finaux possibles prédéfinis dans un module SCA (Service Component Architecture).
- Routage de message dans lequel le flux est dynamique, de même que la sélection des points de contact. Les points de contact de service sont sélectionnés à partir d'une source externe, en phase d'exécution.

Sélection de point de contact dynamique

L'environnement d'exécution est doté d'une fonction de routage des messages de demande et de réponse vers une adresse de point de contact identifiée par un élément d'en-tête de message. Cet élément d'en-tête de message peut être mis à jour par des primitives de médiation dans un flux de médiation. L'adresse de

point de contact peut être mise à jour avec des informations d'un registre, d'une base de données ou avec des informations provenant du message même. Le routage des messages de réponse ne s'applique que si la réponse est envoyée par une exportation JAX-WS de service Web.

Pour que l'environnement d'exécution puisse implémenter le routage dynamique sur une demande ou une réponse, la propriété `Utiliser le point de contact dynamique s'il est défini dans l'en-tête de message` doit être définie sur le module SCA. Les développeurs d'intégration peuvent définir la propriété `Utiliser le point de contact dynamique s'il est défini dans l'en-tête de message` ou la promouvoir (la rendre visible en phase d'exécution) de telle sorte que l'administrateur d'exécution puisse la définir. Vous pouvez visualiser les propriétés de module dans la fenêtre Propriétés de module. Pour afficher la fenêtre, cliquez sur **Applications** > **Modules SCA** > **Propriétés de module**. Le développeur d'intégration donne aux propriétés promues des noms d'alias qui sont affichés sur la console d'administration.

Registre

Vous pouvez utiliser IBM WebSphere Service Registry and Repository (WSRR) pour stocker les informations de point de contact de service puis créer les modules SCA pour extraire les points de contact du registre WSRR.

Lorsque vous développez des modules SCA, vous utilisez la primitive de médiation Recherche de noeud final pour permettre au flux de médiation d'effectuer une requête sur un registre WSRR pour un noeud final de service ou un ensemble de noeuds finaux de service. Si un module SCA extrait un ensemble de points de contact, il doit alors utiliser une autre primitive de médiation pour sélectionner le point de contact à privilégier.

Contrôle des règles de médiation des demandes de service :

Vous pouvez utiliser des règles de médiation pour contrôler les flux de médiation entre les demandeurs de services et les fournisseurs de services.

Vous pouvez contrôler les flux de médiation en utilisant les règles de médiation stockées dans IBM WebSphere Service Registry and Repository (WSRR). L'implémentation de la gestion des règles de service dans WSRR est basée sur Web Services Policy Framework (WS-Policy).

Pour contrôler les demandes de service en utilisant des règles de médiation, vous devez disposer de modules SCA (Service Component Architecture) et de documents de règles de médiation adaptés dans votre registre WSRR.

Savoir connecter une règle de médiation à une demande de service

Lorsque vous développez un module SCA nécessitant l'utilisation d'une règle de médiation, vous devez inclure une primitive de médiation Résolution de règle dans le flux de médiation. En phase d'exécution, la primitive de médiation Résolution de règle obtient des informations sur la règle de médiation à partir du registre. C'est pourquoi, un module SCA doit contenir un composant de flux de médiation afin de prendre en charge le contrôle des règles de médiation des demandes de service.

Dans le registre, vous pouvez associer une ou plusieurs règles de médiation à un module SCA ou à un service utilisé par le module SCA. Les règles de médiation associées peuvent être utilisées (sont dans la portée) pour tous les messages de service traités par ce module SCA. Les règles de médiation peuvent être associées à des connexions de règles qui définissent des conditions. Les conditions de règle de médiation permettent à différentes règles de médiation de s'appliquer dans différents contextes. En outre, les règles de médiation peuvent comporter des classifications utilisables pour indiquer un état de gouvernance.

WebSphere Service Registry and Repository :

WebSphere Service Registry and Repository (WSRR) vous permet de conserver, de consulter et de gérer les informations relatives aux points de contact de service et aux règles de médiation. Vous pouvez utiliser WSRR pour rendre vos applications de service plus dynamiques et plus adaptables à l'évolution des conditions métier.

Introduction

Les flux de médiation peuvent utiliser WSRR comme un mécanisme de recherche dynamique fournissant des informations sur les noeuds de service ou les règles de médiation.

Pour configurer l'accès à WSRR, vous devez créer des documents de définition WSRR via la console d'administration. Vous pouvez aussi utiliser les commandes d'administration WSRR du client de script wsadmin. Les définitions WSRR et leurs propriétés de connexion représentent le mécanisme utilisé pour se connecter à une instance du registre et extraire un point de contact de service ou une règle de médiation.

Noeuds finaux de service

Vous pouvez utiliser WSRR pour conserver des informations sur les services que vous utilisez déjà, que vous prévoyez d'utiliser et dont vous souhaitez avoir connaissance. Ces services peuvent figurer dans vos systèmes ou dans d'autres. Par exemple, une application peut utiliser peut WSRR pour localiser le service le plus à même de répondre à ses besoins fonctionnels et de performances.

Lorsque vous développez un module SCA nécessitant l'accès à des points de contact de service à partir de WSRR, vous devez inclure une primitive de médiation Recherche de point de contact dans le flux de médiation. Pendant la phase d'exécution, celle-ci récupère en effet les points de contact de service du registre.

Règles de médiation

Vous pouvez également utiliser WSRR pour stocker les informations de règle de médiation. Les règles de médiation peuvent vous permettre de contrôler les requêtes de service par substitution dynamique des propriétés du module. Si WSRR contient des règles de médiation associées à un objet représentant votre module SCA ou votre service cible, les règles de médiation peuvent remplacer les propriétés du module. Les conditions de règle de médiation permettent à différentes règles de médiation de s'appliquer dans différents contextes.

Remarque : Les règles de médiation s'occupent du contrôle des flux de médiation mais pas des questions de sécurité.

Lorsque vous développez un module SCA nécessitant l'utilisation d'une règle de médiation, vous devez inclure une primitive de médiation Résolution de règle dans le flux de médiation. En phase d'exécution, la primitive de médiation Résolution de règle obtient des informations sur la règle de médiation à partir du registre.

WebSphere eXtreme Scale :

En utilisant le produit WebSphere eXtreme Scale (eXtreme Scale), vous pouvez fournir un système de mise en cache que vous pouvez intégrer à une application IBM Business Process Manager. En utilisant eXtreme Scale avec IBM Business Process Manager, vous pouvez améliorer les temps de réponse du service et la fiabilité, et fournir des fonctionnalités d'intégration supplémentaires.

eXtreme Scale agit comme une grille de données en mémoire élastique et évolutive. La grille de données met en cache, partitionne, réplique et gère les données et la logique métier de façon dynamique sur de

nombreux serveurs. Avec eXtreme Scale, vous pouvez également obtenir des qualités de service telles que l'intégrité transactionnelle, la haute disponibilité et des temps de réponse prévisibles.

Vous pouvez utiliser des flux de médiation pour accéder à la fonction de mise en cache de eXtreme Scale, , y compris les primitives de médiation WebSphere eXtreme Scale dans votre flux. Lorsque vous développez un module SCA (Service Component Architecture) nécessitant de stocker des informations dans un cache eXtreme Scale, vous devez inclure une primitive de médiation WebSphere eXtreme Scale Stocker dans le flux de médiation. Si vous souhaitez extraire des informations à partir d'un cache eXtreme Scale, vous devez inclure la primitive de médiation WebSphere eXtreme Scale Extraire. En combinant les deux primitives de médiation dans un flux de médiation, vous pouvez mettre en cache la réponse à partir d'un système dorsal, afin que les demandes futures puissent extraire la réponse du cache.

Pour configurer l'accès à eXtreme Scale , vous devez créer une définition WebSphere eXtreme Scale à l'aide de la console d'administration. Vous pouvez aussi utiliser les commandes d'administration WebSphere eXtreme Scale depuis le client de scriptage wsadmin. Une définition eXtreme Scale est le mécanisme utilisé par les primitives de médiation WebSphere eXtreme Scale Extraire et Stocker pour se connecter à un serveur eXtreme Scale.

Message Service Clients

Message Service Clients est disponible pour C/C++ et .NET pour permettre aux applications non-Java de se connecter au bus de service d'entreprise.

Message Service Clients for C/C++ and .NET fournit l'API XMS qui dispose du même groupe d'interfaces que l'API Java Message Service (JMS). Message Service Client for C/C++ contient deux implémentations de XMS, une utilisée par les applications C et une autre utilisée par les applications C++s. Message Service Client for .NET contient une implémentation totalement gérée de XMS, qui peut être utilisée par n'importe quel langage compatible .NET.

Vous pouvez obtenir Message Service Clients for .NET à l'adresse suivante : http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en

Vous pouvez obtenir Message Service Clients for C/C++ à l'adresse suivante : http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Vous pouvez également installer et utiliser le support client Java EE de WebSphere Application Server Network Deployment, y compris les client de services Web, EJB et JMS.

Chapitre 2. En savoir plus sur les concepts clés

Utilisez cette section comme point de départ pour découvrir les technologies utilisées dans et par IBM Business Process Manager.

Scénarios de création

Utilisez des scénarios pour découvrir et utiliser les composants et produits de la famille des produits de gestion des processus métier.

Gestion des versions

Le cycle de vie d'une process application commence par la création de la process application et se poursuit au cours d'un cycle de mise à jour, de déploiement, de co-déploiement, d'annulation du déploiement et d'archivage de la process application. *La gestion des versions* est un mécanisme utilisé pour gérer le cycle de vie de l'application de processus en identifiant de façon unique les versions individuelles de l'application de processus.

La manière dont la gestion de versions fonctionne dans IBM Business Process Manager varie selon ce que vous déployez : une application de processus déployée depuis le référentiel dans IBM Process Center, ou une application d'entreprise déployée directement depuis IBM Integration Designer.

Les applications de processus et les kits d'outils que vous déployez dans un environnement d'exécution à partir de Process Center sont, par défaut, versionnés. Pour les applications d'entreprise, vous pouvez choisir de versionner les modules et les bibliothèques dans IBM Integration Designer.

De plus, vous pouvez créer des versions d'une tâche manuelle ou d'une machine d'état, de sorte que plusieurs versions de la tâche ou de la machine d'état peuvent coexister dans l'environnement d'exécution.

Gestion des versions dans les applications de processus

La gestion de versions permet à l'environnement d'exécution d'identifier les instantanés dans le cycle de vie d'une application de processus et d'exécuter simultanément plusieurs instantanés à la fois sur un serveur de processus.

Pour comprendre la manière dont les versions des applications de processus sont gérées, il est important de rappeler qu'une application de processus est un conteneur composé de plusieurs artefacts utilisés dans ou par l'application de processus (par exemple des modèles de processus ou des BPD, des références de kits d'outils, des services ou des pistes). La gestion des versions s'effectue à ce niveau du conteneur, pas au niveau des artefacts individuels. Pour les applications de processus, cela signifie que la gestion des versions s'effectue lors de la prise d'un instantané.

Vous pouvez comparer des instantanés pour identifier des différences entre les versions. Par exemple, si un développeur a résolu un problème concernant un service et pris un instantané de l'application de processus ou du kit d'outils concerné à ce stade, puis qu'un autre développeur a apporté plusieurs modifications au même service et pris un nouvel instantané, le chef de projet peut comparer ces deux instantanés pour déterminer quelles modifications ont été apportées, quand et par qui. Si le chef de projet décide que les modifications supplémentaires apportées au service ne présentent pas d'intérêt, il peut revenir à l'instantané de la rectification d'origine.

Vous pouvez exécuter simultanément des versions (instantanés) différentes d'une application de processus sur un serveur ; si vous installez un nouvel instantané, supprimez l'original ou laissez son exécution se poursuivre.

Contexte de version

Chaque instantané dispose de métadonnées uniques permettant d'identifier la version (aussi appelée "contexte de version"). Vous affectez cet identificateur, mais IBM recommande d'utiliser un système de version numérique à trois chiffres dans le format <majeur>.<mineur>.<maintenance>. Voir les rubriques sur les conventions de dénomination pour une description plus détaillées de ce schéma de gestion des versions.

IBM Business Process Manager affecte un espace de nom global pour chaque application de processus. L'espace de nom global correspond soit à l'aide de l'application de processus, soit à un instantané spécifique de l'application de processus. Le nom de version utilisé par le serveur ne doit pas dépasser sept caractères, le nom assigné est donc un acronyme qui utilise des caractères issus du nom de l'instantané que vous avez affecté. Les acronymes des instantanés sont identiques aux noms d'instantanés si ces derniers respectent le style IBM VRM recommandé et qu'ils ne dépassent pas sept caractères. Par exemple, le nom d'instantané 1.0.0 a l'acronyme 1.0.0, le nom d'instantané 10.3.0 l'acronyme 10.3.0. L'acronyme d'instantané est garanti unique dans le contexte de l'application de processus, dans la portée du serveur Process Center. C'est pourquoi vous ne pouvez pas modifier l'acronyme d'instantané.

Remarques relatives à la gestion des versions pour les applications de processus sur plusieurs clusters

Vous pouvez installer la même version d'une application de processus sur plusieurs clusters dans une même cellule. Pour faire une différence entre plusieurs installations d'une même version de l'application de processus, créez un instantané de chaque installation et incluez un ID unique de cellule dans le nom de l'instantané (par exemple : v1.0_cell1_1 et v1.0_cell1_2). Chaque instantané est une nouvelle version de l'application de processus (du point de vue de la gestion pure du cycle de vie), mais son contenu et ses fonctions sont identiques.

Lorsque vous installez une application de processus dans un cluster, une synchronisation automatique des noeuds se produit.

Remarques relatives à la gestion des versions pour les kits d'outils Process Designer

N'oubliez pas que les instantanés d'application de processus sont généralement pris lorsque vous êtes prêt à effectuer un test ou une installation. Toutefois, les instantanés d'un kits d'outils sont généralement pris lorsque vous êtes prêt à ce que ce kit d'outils soit utilisé par les applications de processus. Ensuite, pour mettre à jour le toolkit, vous devez prendre un autre instantané de la version actuelle ; les propriétaires des applications de processus et des toolkits peuvent alors décider s'ils veulent passer au nouvel instantané.

Gestion de versions de modules et de bibliothèques

Si un module ou une bibliothèque est contenu dans une application de processus ou un kit d'outils, il adopte le cycle de vie de l'application de processus ou du kit d'outils (versions, instantanés, pistes, etc.). Les noms de bibliothèque et de module doivent être uniques dans toute l'application de processus ou le kit d'outils.

Cette rubrique décrit la gestion de versions des modules et des bibliothèques qui sont utilisés avec les applications de processus. Notez que, si vous déployez des modules directement depuis IBM Integration

Designer sur un Process Server, vous pouvez toujours affecter des numéros de version aux modules au cours du déploiement, comme indiqué dans la rubrique «Création de modules et de bibliothèques versionnés».

Un module ou une bibliothèque associé au IBM Process Center doit trouver ses bibliothèques dépendantes dans la même application de processus ou dans un kit d'outils dépendant.

Le tableau qui suit répertorie les sélections que vous pouvez exécuter dans l'éditeur de dépendances de IBM Integration Designer quand une bibliothèque est associée à une application de processus ou à un kit d'outils.

Tableau 36. Dépendances pour les bibliothèques Module, Application de processus ou kit d'outils et Global

Portée de la bibliothèque	**Description**	**Peut dépendre de ... . .**
Module	Il existe une copie de cette bibliothèque sur le serveur pour chaque module qui l'utilise.	Une bibliothèque ayant pour portée un module peut dépendre de tous les types de bibliothèques.
Application de processus ou kit d'outils	La bibliothèque est partagée entre tous les modules compris dans l'application de processus ou le kit d'outils. Ce paramètre prend effet si le déploiement est réalisé via le IBM Process Center. Si le déploiement a lieu en dehors du IBM Process Center, la bibliothèque est copiée dans chaque module. **Remarque :** Les bibliothèques créées dans IBM Integration Designer version 8 utilisent le niveau de partage **Application de processus ou kit d'outils** par défaut.	Une bibliothèque de ce type peut uniquement dépendre de bibliothèques globales.
Global	La bibliothèque est partagée entre tous les modules en cours d'exécution.	Une bibliothèque globale ne peut dépendre que d'autres bibliothèques globales. **Remarque :** Vous devez configurer une bibliothèque partagée WebSphere pour déployer la bibliothèque globale. Voir «Dépendances de module et de bibliothèque» pour plus d'informations.

Modules et des bibliothèques associés aux applications de processus ou kits d'outils

Il est inutile de versionner les modules et les bibliothèques associés aux applications de processus ou aux kit d'outils.

Les modules et les bibliothèques associés à une application de processus ou un kit d'outils n'ont pas besoin d'être versionnés. En fait, vous ne pouvez pas créer une version d'un module ou d'une bibliothèque avec une application de processus ou un kit d'outils dans l'éditeur de dépendance. Les modules et les bibliothèques associés à une application de processus ou un kit d'outils utilisent des images instantanées, une fonction de Process Center, pour obtenir le même résultat qu'une version.

Les bibliothèques associées à une application de processus ou un kit d'outils n'ont pas de numéro de version nécessaires dans la section Bibliothèques de l'éditeur de dépendance, car aucune version n'est nécessaire.

Conventions de dénomination

Une convention de dénomination permet de différencier les différentes versions d'une application de processus au cours des étapes de son cycle de vie que sont la mise à jour, le déploiement, le codéploiement, l'annulation de déploiement et l'archivage.

La présente section fournit les conventions qui permettent l'identification unique des versions d'une application de processus.

Un *contexte de version* est une combinaison d'acronymes décrivant de façon unique une application de processus ou un kit d'outils. Chaque type d'acronyme est doté d'une convention de dénomination. Le signe est limité à un maximum de sept caractères parmi le jeu de caractères [A à Z et 0 à 9_], hormis pour le sigle de l'instantané qui peut comprendre un point en plus.

- Le sigle de l'application de processus est créé lors de la création de l'application de processus. Il peut comporter au maximum sept caractères.
- L'acronyme de l'instantané est créé automatiquement lors de la création de cet instantané. Il peut comporter au maximum sept caractères.

  Si le nom de l'instantané répond aux critères de validité applicables aux sigles d'instantané, le nom et le sigle de l'instantané seront identiques.

  Remarque : Lors de l'utilisation de la fonction de routage adaptée à la version du composant de flux de médiation, attribuez à votre instantané un nom dans le format <*version*>.<*édition*>.<*modification*> (par exemple, **1.0.0**). Dans la mesure où le sigle de l'instantané est limité à sept caractères, les valeurs numériques sont limitées à un maximum de cinq chiffres au total (cinq chiffres plus deux points). Soyez donc attentif lors de l'incrémentation des zones numériques, dans la mesure où tout ce qui dépasse les sept premiers caractères est tronqué.

  Par exemple, le nom d'instantané **11.22.33** donne le sigle d'instantané **11.22.3**.
- Le sigle de la piste est automatiquement généré à partir du premier caractère de chaque mot composant le nom de piste. Par exemple, une nouvelle piste créée sous le nom **My New Track** donne la valeur de sigle **MNT**.

  La valeur par défaut du nom et du sigle de la piste est **Main**. Le déploiement sur un serveur IBM Process Center inclut le sigle de piste dans le contexte de gestion de versions si ce sigle de piste est différent de **Main**.

Une définition de processus métier dans une application de processus est généralement identifiée par le sigle du nom de l'application de processus, le sigle de l'instantané et le nom de la définition de processus métier. Choisissez des noms uniques pour vos définitions de processus métier chaque fois que cela s'avère possible. S'il existe des noms en double, vous pouvez rencontrer les problèmes suivants :

- Vous ne pourrez peut-être pas exposer les définitions de processus métier en tant que services Web sans une certaine forme de médiation.
- Vous ne pourrez peut-être pas appeler une définition de processus métier créée dans IBM Process Designer à partir d'un processus BPEL créé dans IBM Integration Designer.

Le contexte de version varie selon la façon dont l'application de processus est déployée.

Conventions de dénomination pour les déploiements de serveur Process Center

Sur le serveur IBM Process Center, vous pouvez déployer un instantané d'une application de processus ainsi qu'un instantané d'un kit d'outils. De plus, vous pouvez déployer le tip d'une application de processus ou celui d'un kit d'outils. (Le terme *tip* désigne ici la version opérationelle actuelle d'une application de processus ou d'un kit d'outils). Le contexte de version varie selon le type de déploiement.

Pour les applications de processus, le tip de l'application de processus ou l'instantané d'application de processus sert à identifier de manière unique la version.

Les kits d'outils peuvent être déployés avec une ou plusieurs applications de processus mais le cycle de vie de chaque kit d'outils est lié au cycle de vie de l'application de processus. Chaque application de processus possède sa propre copie du ou des kits d'outils dépendants qui est déployée sur le serveur. Un kit d'outils déployé n'est pas partagé entre les applications de processus.

Si la piste associée à l'application de processus a un autre nom que la valeur par défaut **Main**, le sigle de la piste fait également partie du contexte de la version.

Pour plus d'informations, voir la section «Exemples», à la page 33 plus loin dans cette rubrique.

Instantanés d'application de processus

Pour les déploiements d'instantané d'application de processus, le contexte de la version combine les éléments suivants :

- Sigle du nom de l'application de processus
- Sigle de la piste de l'application de processus (si la piste utilisée n'est pas **Main**)
- Sigle de l'instantané d'application de processus

Kits d'outils autonomes

Pour les déploiements d'instantané de kit d'outils, le contexte de la version combine les éléments suivants :

- Sigle du nom du kit d'outils
- Sigle de la piste du kit d'outils (si la piste utilisée n'est pas **Main**)
- Sigle de l'instantané du kit d'outils

Tips

Les tips d'application de processus sont utilisés au cours des tests itératifs dans Process Designer. Ils peuvent être déployés uniquement sur des serveurs Process Center.

Pour les déploiements de tip d'application de processus, le contexte de la version combine les éléments suivants :

- Sigle du nom de l'application de processus
- Sigle de la piste de l'application de processus (si la piste utilisée n'est pas **Main**)
- "Tip"

Les tips de kit d'outils sont utilisés au cours des tests itératifs dans Toolkit Designer. Ils ne sont pas déployés sur un serveur de production.

Pour les déploiements de tip de kit d'outils, le contexte de la version combine les éléments suivants :

- Sigle du nom du kit d'outils
- Sigle de la piste du kit d'outils (si la piste utilisée n'est pas **Main**)
- "Tip"

Exemples

Les ressources doivent être nommées de manière unique et identifiées de façon externe avec le contexte de la version.

- Le tableau suivant contient un exemple de noms identifiés de manière unique. Dans cet exemple, un tip d'application de processus utilise le nom de piste par défaut (**Main**) :

Tableau 37. Tip d'application de processus avec le nom de piste par défaut

Type de nom	**Exemple**
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Main**
Sigle de la piste d'application de processus	"" (quand la piste est **Main**)
Instantané d'application de processus	
Sigle de l'instantané d'application de processus	**Tip**

Tous les modules SCA associés à ce tip d'application de processus contiennent le contexte de version, comme dans le tableau suivant :

Tableau 38. Modules SCA et fichiers EAR avec version

Nom du module SCA	**Nom avec version**	**Nom de fichier EAR/application avec version**
M1	PA1-Tip-M1	PA1-Tip-M1.ear
M2	PA1-Tip-M2	PA1-Tip-M2.ear

- Le tableau suivant contient un exemple de tip d'application de processus qui utilise un nom de piste autre que le nom par défaut :

Tableau 39. Tip d'application de processus sans le nom de piste par défaut

Type de nom	**Exemple**
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Track1**
Sigle de la piste d'application de processus	**T1**
Instantané d'application de processus	
Sigle de l'instantané d'application de processus	**Tip**

Tous les modules SCA associés à ce tip d'application de processus contiennent le contexte de version, comme dans le tableau suivant :

Tableau 40. Modules SCA et fichiers EAR avec version

Nom du module SCA	**Nom avec version**	**Nom de fichier EAR/application avec version**
M1	PA1-T1-Tip-M1	PA1-T1-Tip-M1.ear
M2	PA1-T1-Tip-M2	PA1-T1-Tip-M2.ear

Des conventions de dénomination similaires s'appliquent aux déploiements avancés d'instantanés et d'astuces du kit d'outils. Elles s'appliquent également aux instantanés avancés installés sur le serveur de processus.

- Le tableau suivant contient un exemple de noms identifiés de manière unique. Dans cet exemple, un instantané d'application de processus utilise le nom de piste par défaut (**Main**):

Tableau 41. Instantané d'application de processus avec un nom de piste par défaut

Type de nom	**Exemple**
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Main**

Tableau 41. Instantané d'application de processus avec un nom de piste par défaut (suite)

Type de nom	Exemple
Sigle de la piste d'application de processus	"" (quand la piste est **Main**)
Instantané d'application de processus	**Process Shapshot V1**
Sigle de l'instantané d'application de processus	**PSV1**

Les modules SCA associés à cet instantané d'application de processus incluent le contexte de version, comme indiqué dans le tableau suivant :

Tableau 42. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-PSV1-M1	PA1-PSV1-M1.ear
M2	PA1-PSV1-M2	PA1-PSV1-M2.ear

- Le tableau suivant montre un exemple d'instantané d'application de processus n'utilisant pas un nom de piste par défaut :

Tableau 43. Instantané d'application de processus sans nom de piste par défaut

Type de nom	Exemple
Nom de l'application de processus	**Process Application 1**
Sigle du nom de l'application de processus	**PA1**
Piste de l'application de processus	**Track1**
Sigle de la piste d'application de processus	**T1**
Instantané d'application de processus	**Process Snapshot V1**
Sigle de l'instantané d'application de processus	**PSV1**

Les modules SCA associés à cet instantané d'application de processus incluent le contexte de version, comme indiqué dans le tableau suivant :

Tableau 44. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-T1-PSV1-M1	PA1-T1-PSV1-M1.ear
M2	PA1-T1-PSV1-M2	PA1-T1-PSV1-M2.ear

Conventions de dénomination pour les déploiements de Process Server

Sur le Process Server, vous pouvez déployer un instantané d'une application de processus. Le sigle de l'instantané d'application de processus est utilisé pour identifier de manière unique la version.

Pour les déploiements d'instantané d'application de processus, le contexte de la version combine les éléments suivants :

- Sigle du nom de l'application de processus
- Sigle de l'instantané d'application de processus

Les ressources doivent être nommées de manière unique et identifiées de façon externe avec le contexte de la version. Le tableau suivant contient des noms identifiés de manière unique :

Tableau 45. Exemples de nom et de sigle

Type de nom	Exemple
Nom de l'application de processus	**Process Application 1**

Tableau 45. Exemples de nom et de sigle (suite)

Type de nom	Exemple
Sigle du nom de l'application de processus	**PA1**
Instantané d'application de processus	**1.0.0**
Sigle de l'instantané d'application de processus	**1.0.0**

Une ressource, par exemple un module ou une bibliothèque, contient le contexte de version dans son identifiant.

Le tableau suivant contient deux modules et illustre comment les fichiers EAR associés incluent le contexte de version.

Tableau 46. Modules SCA et fichiers EAR avec version

Nom du module SCA	Nom avec version	Nom de fichier EAR/application avec version
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

Le tableau suivant contient deux bibliothèques ayant pour portée l'application de processus. Il montre comment les fichiers JAR associés incluent le contexte de version.

Tableau 47. Bibliothèques ayant pour portée l'application de processus et fichiers JAR avec version

Nom de nom de bibliothèque ayant pour portée une application de processus SCA	Nom avec version	Nom de fichier JAR avec version
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Liaisons avec versions

Les applications de processus peuvent contenir des modules SCA contenant des liaisons d'importation et d'exportation. Lorsque vous codéployez des applications, la liaison de chaque version d'une application doit être unique. Certaines liaisons sont automatiquement mises à jour au cours du déploiement pour s'assurer que chacune des versions dispose d'une liaison unique. Dans d'autres cas, vous devez mettre à jour la liaison après le déploiement pour vous en assurer.

Une liaison *spécifique à une version* recouvre une version particulière d'une application de processus, ce qui garantit que chaque liaison est unique pour chacune des applications de processus. Les sections suivantes décrivent les liaisons qui sont automatiquement mises à jour avec une version spécifique ainsi que les actions que vous devez exécuter pendant la phase d'exécution dans le cas d'une liaison sans version spécifique. Pour plus d'informations sur les éléments à prendre en compte lorsque vous créez des modules, voir «Eléments à prendre en compte lors de l'utilisation de liaisons».

SCA

La cible d'une liaison SCA est automatiquement renommée si les liaisons d'importation et d'exportation du module sont définies dans la même portée d'application de processus.

Dans le cas contraire, un message d'information est consigné dans le fichier journal. Vous devez modifier la liaison d'importation après le déploiement pour changer l'adresse cible du noeud final. Vous pouvez utiliser la console d'administration pour ce faire.

Service Web (JAX-WS or JAX-RPC)

L'adresse cible du noeud final d'une liaison de service Web est automatiquement renommée avec une version spécifique au cours du déploiement quand toutes les conditions suivantes sont réunies :

- Vous avez suivi la convention de nom par défaut pour l'adresse :

  http://*ip*:*port*/*nom_module***Web/sca/***nom_exportation***
- L'adresse de noeud final est une adresse SOAP/HTTP.
- Les liaisons d'importation et d'exportation du module sont définies dans la même portée d'application de processus.

Si ces conditions ne sont pas satisfaites, un message d'information est consigné dans le fichier journal. L'action à exécuter dépend de la manière dont vous déployez l'application de processus.

- Si vous déployez conjointement votre application de processus, vous devez renommer manuellement l'adresse URL du noeud final SOAP/HTTP ou la file d'attente de destination SOAP/JMS de manière qu'elle soit unique pour chacune des versions de l'application de processus. Vous pouvez utiliser la console d'administration après le déploiement pour modifier l'adresse cible du noeud final.
- Si vous déployez une unique version de l'application de processus, vous pouvez ignorer ce message

Pour codéployer un instantané de liaison de service Web JMS/SOAP, l'action à exécuter dépend de la manière dont vous déployez votre application de processus :

- Si l'exportation cible et l'importation sont dans la même application de processus, exécutez les opérations suivantes avant de publier l'application de processus sur le Process Center puis créez l'instantané :
  1. Changez l'adresse URL de noeud final de l'exportation. Vérifiez que la destination et la fabrique de connexions sont différentes.
  2. Changez l'adresse URL de noeud final de l'importation de manière qu'elle soit identique à celle que vous avez spécifiée pour l'exportation à l'étape précédente.
- Si l'exportation cible et l'importation sont dans des applications de processus différents, exécutez les opérations suivantes :
  1. Changez l'adresse URL de noeud final de l'exportation. Vérifiez que la destination et la fabrique de connexions sont différentes.
  2. Publiez l'application de processus sur le Process Center.
  3. Créez l'instantané.
  4. Déployez l'application de processus sur le Process Server.
  5. Utilisez la console d'administration WebSphere pour modifier l'adresse URL de noeud final de l'importation correspondante de manière qu'elle soit identique à celle que vous avez spécifiée pour l'exportation.

HTTP

L'adresse URL de noeud final d'une liaison HTTP est automatiquement renommée avec une version spécifique au cours du déploiement quand toutes les conditions suivantes sont réunies :

- Vous avez suivi la convention de nom par défaut pour l'adresse :

  http(s)://*ip*:*port*/*nom_module***Web/***chemin_contexte_export*
- Les liaisons d'importation et d'exportation du module sont définies dans la même portée d'application de processus.

Si ces conditions ne sont pas satisfaites, un message d'information est consigné dans le fichier journal. L'action à exécuter dépend de la manière dont vous déployez l'application de processus.

- Si vous déployez conjointement votre application de processus, vous devez renommer manuellement l'adresse URL de noeud final de manière qu'elle soit unique pour chacune des versions de l'application de processus. Vous pouvez utiliser la console d'administration après le déploiement pour modifier l'adresse cible du noeud final.
- Si vous déployez une unique version de l'application de processus, vous pouvez ignorer ce message

JMS et JMS générique

Les liaisons JMS génériques et les liaisons JMS système prennent automatiquement en compte la version.

Remarque : Pour les liaisons JMS génériques et les liaisons JMS définies par l'utilisateur, il n'y a pas de renommage automatique des liaisons au cours du déploiement pour que les liaisons soient spécifiques à une version. Dans le cas d'une liaison définie par l'utilisateur, pour différencier les versions de l'application de processus, vous devez renommer les attributs suivants :

- Configuration de noeud final
- File d'attente de destination de réception
- Nom du port d'écoute (si défini)

Définissez la destination d'envoi associée si vous modifiez le noeud final du module cible.

MQ/JMS et MQ

Il n'y a pas de renommage automatique des liaisons MQ/JMS et MQ au cours du déploiement pour qu'elles soient spécifiques à une version.

Pour différencier les versions de l'application de processus, vous devez renommer les attributs suivants :

- Configuration de noeud final
- File d'attente de destination de réception

Définissez la destination d'envoi associée si vous modifiez le noeud final du module cible.

EJB

Il n'y a pas de renommage automatique des liaisons EJB au cours du déploiement pour qu'elles soient spécifiques à une version.

Pour différencier les versions de l'application de processus, vous devez renommer les attributs des noms JNDI.

Notez que vous devez aussi mettre à jour les applications client de manière qu'elles utilisent les nouveaux noms JNDI.

EIS

Un adaptateur de ressources est automatiquement renommé pour référencer la version au cours du déploiement si le nom de ressource par défaut (***ModuleName*App:Adapter Description**) n'a pas été modifié.

Si le nom de ressource par défaut a été modifié, les noms des adaptateurs de ressources doivent être différents pour chaque version de l'application de processus.

Si les noms des adaptateurs de ressources ne sont pas différents, un message d'information est journalisé pendant le déploiement pour vous en informer. Vous pouvez renommer manuellement les adaptateurs de ressources après le déploiement à l'aide de la console d'administration.

Appel dynamique avec version

Vous pouvez configurer des composants de flux de médiation pour acheminer des messages vers des noeuds finaux qui sont déterminés dynamiquement lors de l'exécution. Quand vous créez le module de médiation, vous configurez la recherche de noeud final de façon à utiliser le routage avec version.

Si vous utilisez le style IBM_VRM (*<version>.<édition>.<modification>*) pour l'instantané, vous pouvez exporter le fichier EAR de l'application de processus dans WSRR (WebSphere Service Registry and Repository). Quand vous créez le module de médiation, vous configurez ensuite la recherche de noeud final de façon à utiliser le routage avec version. Par exemple, vous sélectionnez **Renvoyer le noeud final correspondant à la dernière version compatible des services utilisant le module SCA** dans la zone **Règle de correspondance** puis vous sélectionnez **SCA** pour **Type de liaison**.

Les versions suivantes de l'application de processus sont déployées sur le serveur et sont publiées dans WSRR. La recherche de noeud final de l'application de processus appelle dynamiquement la dernière version compatible du noeud final de service.

Notez que vous pouvez également définir la cible dans l'en-tête SMOHeader, et la valeur peut être alors transportée par le message de demande.

Déploiement d'applications de processus contenant des projets et des modules Java

Les applications de processus peuvent contenir des projets Java et des modules Java EE personnalisés. Lorsque vous codéployez des applications, le module Java EE personnalisé de chaque version de l'application doit être unique.

Notez que les modules Java EE personnalisés et les projet Java sont déployés sur un serveur s'ils sont déployés avec un module SCA contenant une dépendance déclarée. Si vous ne sélectionnez pas **Déployer avec le module** (valeur par défaut) quand vous déclarez la dépendance, vous devez déployer le module ou le projet manuellement.

Déploiement d'applications de processus contenant des règles métier et des sélecteurs

Si vous déployez plusieurs versions d'une application de processus qui inclut un composant de sélecteur ou une règle métier, faites attention à la manière dont les métadonnées associées sont utilisées par les versions.

Les métadonnées dynamiques d'une règle métier ou d'un composant de sélecteur sont définies pendant la phase d'exécution par le nom de composant, l'espace de nom cible du composant et le type de composant. Si vous déployez au moins deux versions d'une application de processus qui contient une règle métier ou un sélecteur dans le même environnement d'exécution, elles partageront les mêmes métadonnées de logique de règle (règle métier) ou de routage (sélecteur).

Pour que chaque version de la règle métier ou du composant de sélecteur de l'application de processus utilise ses propres métadonnées dynamiques (logique de règle ou routage), restructurez l'espace de nom cible pour qu'ils soit spécifique à chaque version de l'application de processus.

Objets de configuration

Vous pouvez utiliser les commandes AdminConfig de l'outil d'administration de ligne de commande WebSphere (wsadmin) pour afficher et modifier les propriétés de sécurité et de la base de données dans IBM Business Process Manager.

Le terme *objet de configuration* se réfère à un objet auquel on accède à l'aide des commandes AdminConfig de wsadmin. Pour de plus amples informations, voir Propriétés de configuration de la sécurité.

Architecture de déploiement

L'architecture de déploiement de IBM Business Process Manager est constituée de processus logiciels appelés serveurs, d'unités topologiques référencées comme noeuds et cellules et du référentiel de configuration utilisé pour stocker les informations de configuration.

Cellules

Dans IBM Business Process Manager, les *cellules* sont des regroupements logiques d'un ou plusieurs noeuds dans un réseau réparti.

Une cellule est un concept de configuration, un moyen pour les administrateurs d'associer logiquement des noeuds entre eux. Les administrateurs définissent les noeuds qui constituent une cellule, en fonction de critères spécifiques qui ont une signification dans leur environnement organisationnel.

Les données de configuration d'administration sont stockées dans des fichiers XML. Une cellule conserve des fichiers de configuration maîtres pour chaque serveur de chaque noeud de la cellule. Chaque noeud et chaque serveur possèdent également leurs propres fichiers de configuration locaux. Les modifications apportées à un noeud local ou à un fichier de configuration de serveur sont temporaires, si le serveur appartient à la cellule. Tant qu'elles sont en vigueur, les modifications locales remplacent les configurations de cellule. Les modifications apportées aux fichiers de configuration du serveur maître et du noeud maître au niveau de la cellule remplacent les modifications temporaires effectuées sur le noeud lorsque les documents de configuration de la cellule sont synchronisés avec les noeuds. La synchronisation a lieu lors d'événements désignés (par exemple, au démarrage d'un serveur).

Serveurs

Les serveurs assurent les fonctions centrales de IBM Business Process Manager. Les serveurs de processus permettent d'étendre, ou d'accroître, les capacités d'un serveur d'applications à gérer les modules SCA (Service Component Architecture). D'autres serveurs (gestionnaires de déploiement et agents de noeud) permettent de gérer les serveurs de processus.

Un serveur de processus peut être un *serveur autonome* ou un *serveur géré*. Un serveur géré peut être membre d'un *cluster*. L'ensemble des serveurs gérés, des clusters de serveurs et des autres logiciels intermédiaires constitue un *environnement de déploiement*. Dans un environnement de déploiement, chaque serveur géré ou cluster est configuré pour assurer une fonction particulière dans l'environnement de déploiement (par exemple, hôte de destination, hôte de module d'application ou serveur CEI (Common Event Infrastructure). Un serveur autonome est configuré de façon à assurer toutes les fonctions nécessaires.

Les serveurs assurent le fonctionnement de l'environnement d'exécution des modules SCA des ressources utilisées par ces modules (sources de données, spécifications d'activation et destinations JMS) et des ressources fournies par IBM (destinations de messages, conteneurs Business Process Choreographer et serveurs CEI (Common Event Infrastructure).

Un *agent de noeud* est un agent d'administration qui représente un noeud sur votre système et gère les serveurs de ce noeud. Les agents de noeud surveillent les serveurs d'un système hôte et acheminent les demandes d'administration vers les serveurs. Un agent de noeud est créé lorsqu'un noeud est fédéré sur un gestionnaire de déploiement.

Un *gestionnaire de déploiement* est un agent d'administration qui fournit une vue centralisée de la gestion aux différents serveurs et clusters.

Un serveur autonome est défini par un profil autonome, un gestionnaire de déploiement est défini par un profil de gestionnaire de déploiement et les serveurs gérés sont créés au niveau d'un *noeud géré*, qui est défini par un profil de noeud géré.

Serveurs autonomes

Un serveur autonome fournit un environnement pour le déploiement de modules SCA dans un processus serveur unique. Ce processus serveur inclut, entre autres, une console d'administration, une cible de déploiement, le support de messagerie, le gestionnaire de règles métier et un serveur CEI (Common Event Infrastructure).

Un serveur autonome est simple à configurer. Il est équipé d'une console de démarrage rapide permettant de démarrer et d'arrêter le serveur, ou encore d'ouvrir la galerie d'exemples et la console d'administration. Si vous installez les exemples d'IBM Business Process Manager, puis ouvrez la galerie d'exemples, un exemple de solution est déployé sur le serveur autonome. Vous pouvez explorer les ressources utilisées pour cet exemple dans la console d'administration.

Vous pouvez déployer vos propres solutions sur un serveur autonome, mais celui-ci ne dispose pas de la capacité, de l'évolutivité ni de la robustesse nécessaires dans un environnement de production. L'utilisation d'un environnement de déploiement réseau est préférable en environnement de production.

Il est possible de commencer par utiliser un serveur autonome, puis d'inclure celui-ci dans un environnement de déploiement réseau en le fédérant à une cellule de gestionnaire de déploiement, *sous réserve qu'aucun autre noeud n'ait été fédéré avec cette cellule*. Il n'est pas possible de fédérer plusieurs serveurs autonomes dans une seule cellule. Pour fédérer le serveur autonome, utilisez soit la console d'administration du gestionnaire de déploiement, soit la commande **addNode**. Le serveur autonome ne doit pas être en cours d'exécution lorsque vous le fédérez au moyen de la commande **addNode**.

Un serveur autonome est défini par un profil de serveur autonome.

Clusters

Les clusters sont des groupes de serveurs qui sont gérés ensemble et participent à la gestion de la charge de travail.

Un cluster peut contenir des noeuds ou des serveurs d'applications individuels. Un noeud correspond généralement à un système informatique physique possédant une adresse hôte IP distincte, qui exécute un ou plusieurs serveurs d'applications. Les clusters peuvent être regroupés sous la configuration d'une cellule, qui associe logiquement entre eux de nombreux serveurs et clusters possédant des configurations et des applications différentes en fonction de critères choisis par l'administrateur en adéquation avec ses environnements organisationnels.

Les clusters sont chargés d'équilibrer la charge de travail entre les serveurs. Les serveurs qui appartiennent à un cluster sont appelés membres du cluster. Lorsque vous installez une application sur un cluster, cette application est automatiquement installée sur chaque membre du cluster.

Chaque membre du cluster contenant les mêmes applications, vous pouvez distribuer les tâches client en fonction des capacités des différentes machines, en affectant des pondérations à chaque serveur.

L'affectation de pondérations aux serveurs d'un cluster améliore les performances et la reprise sur incident. Les tâches sont affectées au serveurs qui disposent de la capacité requise pour effectuer les opérations associées. Si un serveur n'est pas disponible pour effectuer la tâche, cette dernière est affectée à un autre membre du cluster. La possibilité de réaffecter les tâches offre des avantages évidents par rapport à l'exécution d'un seul serveur d'applications, qui peut vite être surchargé lorsque le nombre de demandes est trop important.

Profils

Un profil définit un environnement d'exécution unique, associé à des fichiers spécifiques (commandes, configuration et journaux). Les profils définissent trois types d'environnement différents sur les systèmes IBM Business Process Manager : serveur autonome, gestionnaire de déploiement et noeud géré.

Les profils permettent de définir plusieurs environnements d'exécution sur un système sans installer plusieurs copies des fichiers binaires d'IBM Business Process Manager.

Utilisez l'Utilitaire de ligne de commande BPMConfig pour créer des profils IBM BPM. L'utilitaire de ligne de commande **manageprofiles** ou son interface utilisateur graphique, Profile Management Tool (PMT), peuvent être utilisés comme alternative pour créer le profil de gestionnaire de déploiement ou le profil de noeud géré. PMT n'est plus pris en charge pour la création de profils autonomes.

Remarque : Sur les plateformes réparties, chaque profil possède un nom unique. Sous z/OS, tous les profils sont nommés «default». Vous ne pouvez pas renommer, modifier, copier ou supprimer des profils sous z/OS.

Types de profils

Les types de profils IBM BPM suivants sont disponibles avec IBM Business Process Manager version 8.5 :

Profil autonome IBM BPM
Le profil autonome définit des serveurs autonomes avec des capacités et fonctionnalités spécifiques aux configurations d'IBM BPM Express. Vous pouvez créer le profil autonome à l'aide du modèle de profil fourni sous `BPM/BpmServer`, qui est uniquement pris en charge dans IBM BPM Express.

Profil de gestionnaire de déploiement IBM BPM
Le profil de gestionnaire de déploiement définit un gestionnaire de déploiement fournissant une interface d'administration unique à un groupe logique de serveurs sur un ou plusieurs postes de travail. Vous pouvez créer le profil de gestionnaire de déploiement à l'aide du modèle de profil fourni sous `BPM/BpmDmgr`, qui est uniquement pris en charge dans IBM BPM Standard et IBM BPM Advanced.

Profil de noeud géré IBM BPM
Le profil de noeud géré définit un noeud géré lorsque le noeud est fédéré à un gestionnaire de déploiement. Vous pouvez créer le profil de noeud géré à l'aide du modèle de profil fourni sous `BPM/BpmNode`, qui est uniquement pris en charge dans IBM BPM Standard et IBM BPM Advanced.

Les types de profils spécifiques sont disponibles pour chaque configuration IBM BPM.

Tableau 48. Types de profils IBM BPM disponibles

	Types de profils		
Configuration IBM BPM	**Autonome**	**Gestionnaire de déploiement**	**Noeud géré**
IBM BPM Express	Oui	Non	Non
IBM BPM Standard	Non	Oui	Oui
IBM BPM Advanced	Non	Oui	Oui
Environnement UTE (Unit Test Environment) pour IBM Integration Designer	Oui	Facultatif	Facultatif

Répertoire de profil

Chaque profil du système possède son propre répertoire contenant tous ses fichiers. Vous indiquez l'emplacement de ce répertoire lors de la création du profil. Par défaut, il s'agit du répertoire `profiles` dans le répertoire où IBM Business Process Manager est installé. Par exemple, le profil Dmgr se trouve sous `C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr`.

Profil par défaut

Le premier profil créé sur une installation d'IBM Business Process Manager représente le *profil par défaut*. Ce profil est la cible par défaut des commandes émises à partir du répertoire `bin` situé dans le répertoire racine d'installation d'IBM Business Process Manager. Lorsqu'il n'existe qu'un seul profil sur un système, chaque commande fonctionne sur ce profil. Si vous créez un autre profil, vous pouvez faire de ce profil le profil par défaut.

Remarque : Le profil par défaut n'est pas nécessairement un profil portant le nom «par défaut».

Extension de profils

Si vous avez déjà un profil de gestionnaire de déploiement, un profil de noeud géré ou un profil serveur autonome créé pour WebSphere Application Server Network Deployment, vous pouvez l'*étendre* pour prendre en charge IBM Business Process Manager en plus de la fonction existante. Pour étendre un profil, vous devez tout d'abord installer IBM Business Process Manager. Utilisez ensuite l'utilitaire de ligne de commande **manageprofiles** pour étendre un profil autonome ou utilisez le Profile Management Tool ou l'utilitaire de ligne de commande **manageprofiles** pour étendre un profil de gestionnaire de déploiement ou un profil de noeud géré.

Important : Dans un environnement de déploiement réseau, vous devez d'abord étendre le profil de gestionnaire de déploiement, puis les profils de noeud géré.

Restriction : Vous ne pouvez pas étendre un profil autonome ou un profil de gestionnaire de déploiement si le registre d'utilisateur par défaut de WebSphere VMM a été modifié, par exemple, pour utiliser LDAP.

Gestionnaires de déploiement

Un gestionnaire de déploiement est un serveur permettant de gérer les opérations liées à un groupe logique ou à une cellule comprenant d'autres serveurs. Le gestionnaire de déploiement est l'emplacement central permettant d'administrer les serveurs et clusters.

Lors de la création d'un environnement de déploiement, le profil du gestionnaire de déploiement est le premier profil créé. Chaque environnement de déploiement créé possède une console de démarrage rapide permettant de démarrer et arrêter le gestionnaire de déploiement et de démarrer la console d'administration de celui-ci. La console d'administration du gestionnaire de déploiement permet de gérer les serveurs et clusters contenus dans la cellule. Ces opérations incluent la configuration des serveurs et des clusters, l'ajout de serveurs à des clusters et le déploiement de modules SCA.

Le gestionnaire de déploiement est lui-même un serveur, mais vous ne pouvez pas y déployer de modules.

Noeuds

Un *noeud* est un regroupement logique de serveurs gérés.

Un noeud correspond généralement à un système informatique logique ou physique doté d'une adresse hôte IP distincte. Les noeuds ne peuvent pas couvrir plusieurs ordinateurs. Les noms de noeud sont généralement identiques au nom d'hôte de l'ordinateur.

Les noeuds d'une topologie de déploiement réseau peuvent être gérés ou non gérés. Un noeud géré possède un processus d'agent de noeud qui gère sa configuration et ses serveurs. Les noeuds non gérés ne possèdent pas d'agent de noeud.

Noeuds gérés

Un *noeud géré* est un noeud qui est fédéré dans un gestionnaire de déploiement et qui contient un agent de noeud et peut contenir des serveurs gérés. Sur un noeud géré, vous pouvez configurer et exécuter des serveurs gérés.

Les serveurs configurés sur un noeud géré constituent les ressources de votre environnement de déploiement. Les opérations de création, configuration, démarrage, arrêt, gestion et suppression de ces serveurs s'effectuent via la console d'administration du gestionnaire de déploiement.

Une noeud géré est un agent de noeud qui gère tous les serveurs sur un noeud.

Lorsqu'un noeud est fédéré, un processus d'agent de noeud est créé automatiquement. Cet agent de noeud doit être en cours d'exécution pour permettre de gérer la configuration du profil. Par exemple, lorsque vous effectuez les tâches suivantes :

- Démarrer et arrêter les processus serveur.
- Synchroniser les données de configuration sur le gestionnaire de déploiement avec la copie située sur le noeud.

Toutefois, l'agent de noeud ne doit pas nécessairement être en cours d'exécution pour que les applications soient exécutées, ni pour que les ressources soient configurées sur le noeud.

Un noeud géré peut contenir un ou plusieurs serveurs, qui sont administrés par un gestionnaire de déploiement. Vous pouvez déployer des solutions sur les serveurs en mode géré, mais le noeud géré ne contient pas de galerie d'exemples d'applications. Le noeud géré est défini par un profil géré par noeud et possède une console de démarrage rapide.

Noeuds non gérés

Un noeud non géré ne possède pas d'agent de noeud pour gérer ses serveurs.

Des noeuds non gérés dans la topologie de déploiement réseau peuvent avoir des définitions de serveurs, comme des serveurs Web, mais pas de définitions de serveur d'applications. Les noeuds non gérés ne peuvent jamais être fédérés. Cela signifie qu'un agent de noeud ne peut jamais être ajouté à un noeud non géré. Un serveur autonome représente un autre type de noeud non géré. Le gestionnaire de déploiement ne peut pas gérer ce serveur autonome car il n'est pas connu de la cellule. Un serveur autonome peut être fédéré. Lorsqu'il est fédéré, un agent de noeud est automatiquement créé. Le noeud devient un noeud géré dans la cellule.

Agents de noeud

Les agents de noeud sont des agents d'administration qui acheminent les demandes administratives destinées aux serveurs.

Un agent de noeud est un serveur exécuté sur chaque système informatique hôte qui participe à la configuration de Network Deployment. Il s'agit purement d'un agent d'administration, qui n'est pas impliqué dans les fonctions de traitement des applications. Un agent de noeud héberge également d'autres fonctions d'administration importantes, telles que les services de transfert de fichiers, la synchronisation de la configuration et le contrôle des performances.

Remarques relatives aux noms de profils, de noeuds, de serveurs, d'hôtes et de cellules

Cette rubrique indique les termes réservés et les conditions à respecter pour nommer un profil, un noeud, un serveur, un hôte et une cellule (le cas échéant). Cette rubrique s'applique aux plateformes réparties.

Remarques relatives aux noms de profils

Le nom de profil peut être tout nom unique, avec les restrictions suivantes. N'utilisez aucun des caractères suivants :

- Espaces
- Caractères spéciaux non autorisés dans un nom de répertoire sur le système d'exploitation, par exemple *, & ou ?.
- Barres obliques (/) ou barres obliques inversées (\)

Les caractères codés sur deux octets sont autorisés.

Windows **Remarques liées au chemin de répertoire :** Le chemin du répertoire d'installation ne doit pas comporter plus de 60 caractères. Le nombre de caractères du répertoire *chemin_répertoire_profils\nom_profil* doit être inférieur ou égal à 80 caractères.

Remarque : Utilisez une convention de dénomination de chemin court lorsque vous créez un profil dans l'environnement Windows pour éviter la limitation à 255 caractères des chemins Windows.

Remarques relatives aux noms de noeuds, de serveurs, d'hôtes et de cellules

Noms réservés : Evitez d'utiliser des noms réservés comme valeurs de zones. En effet, l'utilisation de noms réservés peut entraîner des résultats imprévisibles. Les mots suivants sont réservés :

- cellules
- noeuds
- serveurs
- clusters
- applications
- déploiements

Descriptions des zones figurant dans les pages Noms de noeud et d'hôte et Noms de noeud, d'hôte et de cellule : suivez les instructions d'attribution de noeud appropriées lorsque vous créez les profils.

- Profils de serveur autonomes
- Profils du gestionnaire de déploiement
- Profils de noeud géré

Tableau 49. Instructions d'attribution de nom pour les profils de serveur autonomes

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom du noeud	Linux UNIX Windows *shortHostName* Node *NodeNumber* où : • *shortHostName* représente le nom d'hôte abrégé. • *NodeNumber* est un nombre séquentiel commençant, séquence commençant à partir de 01.	N'utilisez pas de nom réservé.	Sélectionnez le nom de votre choix. Si vous envisagez de créer plusieurs serveurs sur le même système, choisissez un nom unique afin de simplifier l'installation.
Nom du serveur	Linux UNIX Windows server1	Utilisez un nom unique pour le serveur.	Nom logique du serveur.

Tableau 49. Instructions d'attribution de nom pour les profils de serveur autonomes (suite)

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom d'hôte	Linux UNIX Windows Nom long du serveur DNS (Domain Name Server).	Utilisez un nom de système hôte qualifié complet dont l'adresse peut être résolue sur le réseau.	Utilisez le nom DNS ou l'adresse IP du poste de travail pour permettre la communication avec ce dernier. Consultez les informations supplémentaires sur le nom d'hôte, à la suite de ce tableau.

Tableau 50. Instructions d'attribution de nom pour les profils de gestionnaire de déploiement

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom du noeud	Linux UNIX Windows *shortHostName* Cell Manager*Node Number* où : • *shortHostName* représente le nom d'hôte abrégé. • *NodeNumber* est un nombre séquentiel commençant, séquence commençant à partir de 01.	Utilisez un nom unique pour le gestionnaire de déploiement. N'utilisez pas de nom réservé.	Le nom est utilisé à des fins d'administration dans la cellule de gestionnaire de déploiement.
Nom d'hôte	Linux UNIX Windows Nom long du serveur DNS (Domain Name Server).	Utilisez un nom de système hôte qualifié complet dont l'adresse peut être résolue sur le réseau. N'utilisez pas de nom réservé.	Utilisez le nom DNS ou l'adresse IP du poste de travail pour permettre la communication avec ce dernier. Consultez les informations supplémentaires sur le nom d'hôte, à la suite de ce tableau.

Tableau 50. Instructions d'attribution de nom pour les profils de gestionnaire de déploiement (suite)

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom de la cellule	Linux UNIX Windows *shortHostName* Cell *CellNumber* où : • *shortHostName* représente le nom d'hôte abrégé. • *CellNumber* représente un chiffre séquentiel commençant par 01.	Utilisez un nom unique pour la cellule du gestionnaire de déploiement. Un nom de cellule doit être unique dans tous les cas où le produit s'exécute sur le même poste de travail physique ou le même cluster de postes de travail, par exemple un Sysplex. En outre, il doit être unique dès lors que la connectivité réseau entre entités est requise entre les cellules ou à partir d'un client devant communiquer avec chacune des cellules. Les noms de cellule doivent également être uniques si les espaces noms associés sont sur le point d'être fédérés. Si cette condition n'est pas respectée, des erreurs de type javax.naming.Name NotFoundException peuvent survenir, au quel cas vous devez créer des cellules avec des noms uniques.	Tous les noeuds fédérés deviennent membres de la cellule du gestionnaire de déploiement définie dans la page des noms de noeud, d'hôte et de cellule de l'outil de gestion des profils.

Tableau 51. Instructions d'attribution de nom pour les profils de noeud géré

Nom de la zone	Valeur par défaut	Restrictions	Description
Nom du noeud	Linux UNIX Windows *shortHostName* Node *NodeNumber* où : • *shortHostName* représente le nom d'hôte abrégé. • *NodeNumber* est un nombre séquentiel commençant, séquence commençant à partir de 01.	N'utilisez pas de nom réservé. Utilisez un nom unique dans la cellule du gestionnaire de déploiement.	Ce nom est utilisé à des fins d'administration dans la cellule de gestionnaire à laquelle le profil de noeud géré est ajouté. Utilisez un nom unique dans la cellule du gestionnaire de déploiement.
Nom d'hôte	Linux UNIX Windows Nom long du serveur DNS (Domain Name Server).	Utilisez un nom de système hôte qualifié complet dont l'adresse peut être résolue sur le réseau.	Utilisez le nom DNS ou l'adresse IP du poste de travail pour permettre la communication avec ce dernier. Consultez les informations supplémentaires sur le nom d'hôte, à la suite de ce tableau.

Remarques concernant le nom d'hôte :

Le nom d'hôte correspond au nom réseau du poste de travail physique sur lequel le noeud est installé. Il doit être résolu en noeud réseau physique sur le serveur. Si le serveur contient plusieurs cartes réseau, le nom d'hôte ou l'adresse IP doit être résolu sur l'une d'elles. Les noeuds distants utilisent le nom d'hôte pour se connecter à ce noeud et communiquer avec lui.

IBM Business Process Manager est compatible avec le protocole IP version 4 (IPv4) et version 6 (IPv6). Chaque fois que des adresses IP peuvent être indiquées dans la console d'administration ou via un autre point d'accès, vous pouvez spécifier l'un ou l'autre format. Il est à noter que si le protocole IPv6 est mis en oeuvre sur votre système, vous devez spécifier l'adresse IP suivant ce format. Inversement, si ce protocole n'est pas disponible, entrez les adresses IP au format IPv4. Pour plus d'informations sur IPv6, reportez-vous à la description suivante : IPv6.

Les instructions suivantes peuvent aider à déterminer le nom d'hôte approprié à appliquer à votre poste de travail :

- Sélectionnez un nom d'hôte accessible via les autres postes de travail du réseau.
- N'utilisez pas l'identificateur générique 'localhost' pour cette valeur.
- Ne tentez pas d'installer les produits IBM Business Process Manager sur un serveur portant un nom d'hôte qui utilise des caractères DBCS (Double-Byte Character Set). En effet, les caractères DBCS ne sont pas pris en charge lorsqu'ils sont utilisés dans le nom d'hôte.
- Evitez d'utiliser le trait de soulignement (_) dans les noms de serveurs. Les normes Internet exigent que les noms de serveurs soient conformes aux normes décrites dans les documents Internet Official Protocol Standards RFC 952 et RFC 1123. Les noms de domaines ne doivent contenir que des lettres (en majuscules et en minuscules) et des chiffres. Les noms de domaines peuvent également contenir des tirets (-), sous réserve de ne pas se trouver en fin de nom. Les traits de soulignement (_) ne sont pas acceptés dans le nom d'hôte. Si vous avez installé IBM Business Process Manager sur un serveur dont le nom comporte un trait de soulignement, vous devez accéder à ce serveur au moyen de son adresse IP jusqu'à ce que vous l'ayez renommé.

Si vous définissez des noeuds coexistant sur le même système avec des adresses IP uniques, définissez chaque adresse IP dans une table de recherche DNS (Domain Name Server). Les fichiers de configuration des serveurs ne fournissent pas de fonction de résolution du nom de domaine pour les adresses IP définies sur un poste de travail doté d'une adresse réseau unique.

La valeur indiquée pour le nom d'hôte est utilisée pour la propriété hostName dans les documents de configuration. Indiquez la valeur du nom d'hôte dans l'un des formats suivants :

- Chaîne représentant le nom d'hôte DNS (Domain Name Server) complet, tel que xmachine.manhattan.ibm.com
- Nom d'hôte DNS abrégé par défaut, tel que xmachine
- Adresse IP numérique, telle que 127.1.255.3

Le nom d'hôte DNS complet permet d'éviter toute ambiguïté et est extrêmement souple. Vous avez la possibilité de modifier l'adresse IP réelle du système hôte sans modifier la configuration du serveur. La valeur définie pour le nom d'hôte est particulièrement utile si vous avez l'intention de modifier fréquemment l'adresse IP lorsque vous utilisez DHCP (Dynamic Host Configuration Protocol) pour affecter des adresses IP. L'inconvénient de ce format est qu'il dépend d'un serveur DNS. Si le serveur DNS n'est pas disponible, la connectivité est compromise.

Le nom d'hôte peut également être résolu de manière dynamique. En outre, le format de nom court étant redéfini dans le fichier hosts local, le système peut exécuter le serveur, même si ce dernier est déconnecté du réseau. Associez le nom abrégé à la valeur 127.0.0.1 (boucle locale) dans le fichier hosts pour lancer l'exécution en étant déconnecté. L'inconvénient du format de nom abrégé est qu'il dépend d'un serveur DNS pour l'accès distant. Si le serveur DNS n'est pas disponible, la connectivité est compromise.

Dans ce dernier cas, la résolution du nom via DNS n'est pas nécessaire. Un noeud distant peut se connecter à l'hôte désigné par une adresse IP sans avoir recours au serveur DNS. L'inconvénient de ce format est que l'adresse IP numérique est fixe. Vous devez modifier la propriété hostName dans les fichiers de configuration lorsque vous modifiez l'adresse IP du poste de travail. Par conséquent, n'utilisez pas d'adresse IP si vous utilisez le protocole DHCP (Dynamic Host Configuration Protocol) ou si vous changez souvent d'adresse IP. En outre, vous ne pouvez pas utiliser le noeud si l'adresse IP de l'hôte est déconnectée du réseau.

BPMN 2.0

IBM Business Process Manager prennent en charge la sous-classe Common Executable de la classe de conformité de BPMN 2.0 Process Modeling, qui traite des modèles exécutables.

BPMN (Business Process Model and Notation) est la norme de base des processus dans IBM Process Designer et IBM Process Center. Les diagrammes de définition de processus métier reposent sur la spécification BPMN. Cette rubrique présente certaines des méthodes par lesquelles BPMN 2.0 est appliqué dans IBM Business Process Manager. Pour obtenir des informations sur BPMN, reportez-vous à la page des spécifications BPMN à l'adresse suivante http://www.bpmn.org/.

IBM Business Process Manager prend en charge les types de tâche BPMN 2.0 suivants :

- Aucun (tâche abstraite dans la spécification BPMN 2.0)
- Tâche système (tâche de maintenance dans la spécification BPMN 2.0)
- Tâche utilisateur
- Script
- Tâche de décision (tâche de règle métier dans la spécification BPMN 2.0)

Les événements de message intermédiaires IBM BPM fournissent des fonctionnalités similaires à la tâche d'envoi et de réception BPMN.

Notation BPMN 2.0

A partir de la V7.5.1, les nouvelles icônes Process Designer de BPMN 2.0 des diagrammes de BPD sont rassemblées dans une palette simplifiée et affichées dans les diagrammes de processus. Ces icônes indiquent si votre activité est une tâche système, une tâche utilisateur, une tâche de décision, un script ou un processus lié. Les activités des modèles créés dans des versions antérieures affichent également les types de tâches et les icônes de tâches BPMN 2.0 appropriées lorsque vous les affichez dans la version 7.5.1 ou ultérieure.

Activités et tâches

Certaines modifications de terminologie ont été effectuées par rapport aux versions précédentes de Process Designer. Un certain nombre d'entre elles concernent des types d'activités qui ont été renommés.

- Les activités de maintenance (automatisées) sont à présent des tâches système.
- Les activités de maintenance (tâche) d'un couloir non système sont à présent des tâches utilisateur.
- Les activités de maintenance (tâche) d'un couloir système sont à présent des tâches de décision si elles font référence à un service de décision.
- Les activités de maintenance (tâche) d'un couloir système sont à présent des tâches système si elles font référence à un type de service autre qu'un service de décision.
- Les activités Javascript sont à présent des tâches de script.
- Les activités de processus imbriquées sont à présent des processus liés.
- Les activités externes provenant des versions précédentes de Process Designer sont disponibles en tant qu'implémentations externes pour les tâches utilisateur ou les tâches système.

Passerelles

Il n'existe pas de changements de notation concernant les passerelles des versions précédentes. Il existe toutefois trois changements de terminologie. La passerelle de décision est à présent la *passerelle exclusive*, la passerelle de jointure/de division simple est à présent la *passerelle parallèle* et la passerelle de jointure/de division conditionnelle est à présent la *passerelle inclusive*.

Il existe également un nouveau type de passerelle, appelé la *passerelle d'événement*. Une passerelle de branchement représente un point de branchement dans un processus dans lequel les chemins de substitution qui suivent la passerelle sont basés sur des événements qui se produisent, plutôt que sur l'évaluation des expressions utilisant des données de processus (comme avec une passerelle exclusive ou inclusive). Un événement spécifique, généralement le reçu d'un message, détermine le chemin qui sera pris.

Evénements sans interruption

BPMN 2.0 a ajouté une notation pour les événements sans interruption. Par défaut, un événement limite interrompt l'activité à laquelle il est attaché. Lorsque l'événement est déclenché, l'activité est arrêtée et le jeton poursuit son chemin dans le flux de séquence sortant de l'événement. Si l'événement est défini comme sans interruption, lorsqu'il est déclenché, l'activité attachée continue en parallèle et un nouveau jeton est généré puis transmis au flux de séquence sortant de l'événement. La limite de l'événement se transforme en ligne en pointillés pour les événements sans interruption.

Les événements intermédiaires attachés aux activités sont à présent des événements intermédiaires d'interruption s'ils ferment leurs activités attachées, ou des événements intermédiaires sans interruption dans le cas contraire.

Evénement de démarrage

La spécification BPMN permet aux modèles de processus d'omettre les symboles d'événement de démarrage et de fin. Process Designer nécessite que les modèles de processus utilisent des événements de démarrage et de fin.

Trois types d'événements de début sont disponibles dans Process Designer :

processus

- aucun
- message
- ad hoc

sous-processus

- aucun

sous-processus d'événement

- erreur
- message
- minuteur

Vous pouvez modifier le type d'un événement de démarrage en éditant les propriétés de l'événement. Un processus peut comporter plusieurs événements de démarrage de message mais vous ne pouvez utiliser qu'un seul événement de démarrage Aucun.

Les événements de fin

Il existe quatre types d'événement de fin disponibles : *message*, *terminer*, *erreur* et *aucun*. Vous pouvez changer le type d'un événement de fin.

Lorsqu'un processus parent appelle un processus enfant et que ce processus enfant exécute une action d'événement Terminer, le processus enfant s'arrête et le processus parent passe alors aux étapes suivantes.

Sous-processus

La spécification BPMN définit deux types de sous-processus, imbriqué et réutilisable. Vous pouvez créer les deux types dans Process Designer. Les sous-processus imbriqués sont appelés simplement *sous-processus* dans Process Designer et sont nouveaux dans la version 7.5.1. Le sous-processus réutilisable BPMN est appelé *processus lié* dans Process Designer.

Un sous-processus existe dans le processus contenant. Il permet de rassembler des étapes du processus dans le but de réduire la complexité et l'encombrement des diagrammes. Les sous-processus regroupent plusieurs étapes en une seule activité. Le sous-processus ne peut être vu que par le processus dans lequel il est défini. Un sous-processus existe dans l'étendue de son appelant et il a accès à toutes les variables contenues dans cet environnement. Il n'existe aucun paramètre transmis à partir de ou à destination du sous-processus intégré.

En dehors du sous-processus et du processus lié, Process Designer comporte un sous-processus d'événement, qui est un sous-processus spécialisé utilisé pour la gestion des événements. Il n'est pas relié à d'autres activités via le flux de séquences et est exécuté uniquement si son événement de démarrage est déclenché.

Processus liés

Un sous-processus réutilisable BPMN est appelé *processus lié* dans Process Designer. Il s'agit d'un processus créé en dehors du processus en cours pouvant être appelé par le processus actuel. IL est réutilisable car les autres définitions de processus peuvent également appeler ce processus. Le processus lié définit ses paramètres d'entrée et de sortie et il n'a pas accès à la portée ou à l'environnement de l'appelant. Le processus lié est similaire au processus imbriqué qui était disponible dans les anciennes versions ; aucun changement de comportement de l'activité n'est à noter. Les processus imbriqués existants sont migrés vers les processus liés. Le processus lié ressemble à un sous-processus avec une limite épaisse et il est mis en évidence dans la fenêtre Inspecteur.

Boucles

BPMN offre le concept d'une activité pouvant être répétée. L'activité peut être atomique, ce qui signifie qu'elle peut être répétée, ou elle peut être un sous-processus encapsulant une série d'étapes qui sont répétées. Si vous développez l'activité répétée, vous voyez les activités contenues devant être exécutées de façon répétitive. La condition est toujours évaluée au début de chaque itération de boucle. Il n'existe pas de possibilité d'évaluation à la fin de chaque itération de boucle.

IBM Business Process Manager possède une *boucle multi-instance*, qui est exécutée un nombre infini de fois avec les activités qu'elle contient de façon séquentielle ou en parallèle.

Importation de processus non-BPMN

Vous pouvez importer des modèles créés dans IBM WebSphere Business Modeler et les utiliser dans Process Designer. Pour plus d'informations sur l'importation BPMN 2.0, voir Mappage des éléments IBM WebSphere Business Modeler vers des constructions IBM Business Process Manager. Vous pouvez également importer des modèles BPMN 2.0 créés dans IBM WebSphere Business Compass, Rational Software Architect ou dans d'autres environnements de modélisation.

Définitions de processus métier (BPD)

Pour modéliser un processus dans IBM Process Designer, vous devez créer une définition de processus métier (BPD). La définition de processus métier peut se baser sur un modèle BPMN importé.

Une définition de processus métier est un modèle réutilisable d'un processus définissant ce qui est commun à toutes les instances d'exécution de ce modèle de processus. Une BPD doit contenir un événement de début, un événement de fin, au moins un couloir et une ou plusieurs activités. Pour plus d'informations sur les limitations de caractères qui s'appliquent au BPD, voir "Conventions de dénomination d'IBM Process Designer" dans les liens connexes.

Une définition de processus métier (BPD, Business Process Definition) doit inclure un couloir pour chaque système ou groupe d'utilisateurs qui participent à un processus. Un couloir peut être un couloir participant ou un couloir système. Toutefois, vous pouvez créer une BPD qui regroupe les activités d'un groupe et d'un système dans un même couloir si cela vous arrange. Pour plus d'informations sur la création d'une BPD, voir "Création d'une définition de processus métier (BPD)" dans les liens connexes.

Vous pouvez désigner n'importe quelle personne ou n'importe quel groupe comme responsable des activités d'un couloir participant. Chaque couloir que vous créez est affecté par défaut au groupe Tous les utilisateurs. Vous pouvez utiliser ce groupe par défaut pour exécuter et tester votre définition de processus métier dans Inspector. Le groupe Tous les utilisateurs inclut tous les utilisateurs membres du groupe de sécurité **tw_allusers**, lequel est un groupe de sécurité spécial incluant automatiquement tous les utilisateurs du système.

Un couloir système contient les activités gérées par un système IBM Process Center spécifique. Chaque activité nécessite une implémentation, qui définit l'activité et les propriétés de la tâche. Lors de l'implémentation, un développeur crée un service ou écrit le code JavaScript nécessaire à l'exécution des activités dans le couloir système. Pour plus d'informations sur les services, voir "Comprendre les types de services" dans les liens connexes.

Pour chaque définition de processus métier que vous créez, vous devez déclarer des variables afin de capturer les données métier qui sont transmises d'une activité à l'autre tout au long du processus. Pour plus d'informations sur l'implémentation des variables, voir "Gestion et mappage de variables" dans les liens connexes.

Vous pouvez également ajouter des événements à une BPD. Les événements présents dans IBM BPM peuvent être déclenchés par un dépassement de date d'exigibilité, par une exception ou par un message entrant. Le déclencheur souhaité détermine le type d'événement à implémenter. Pour plus d'informations sur les types d'événement disponibles et leurs déclencheurs, voir "Modélisation d'événements".

Liaisons

Au coeur de l'architecture orientée services se trouve le concept de *service*, une unité de fonctionnalité accomplie par une interaction entre les périphériques de traitement. Une *exportation* définit l'interface externe (ou le point d'accès) d'un module, de telle sorte que les composants SCA (Service Component Architecture) au sein du module puissent fournir leurs services à des clients externes. Une *importation* définit une interface vers des services situés à l'extérieur d'un module, de telle sorte que les services pussent être appelés depuis le module. Vous pouvez utiliser des *liaisons* spécifiques à un protocole avec des importations et des exportations pour spécifier les moyens de transport des données depuis ou vers le module.

Exportations

Les clients externes peuvent appeler des composants SCA dans un module d'intégration via une variété de protocoles (comme HTTP, JMS, MQ et RMI/IIOP) avec des données dans une variété de formats (comme XML, CSV, COBOL et JavaBeans). Les exportations sont des composants qui reçoivent ces requêtes de sources externes et appellent ensuite des composants IBM Business Process Manager à l'aide du modèle de programmation SCA.

Par exemple, dans la figure suivante, une exportation reçoit une requête via le protocole HTTP d'une application client. Les données sont transformées en un objet métier, le format utilisé par le composant

SCA. Le composant est ensuite appelé avec cet objet de données.

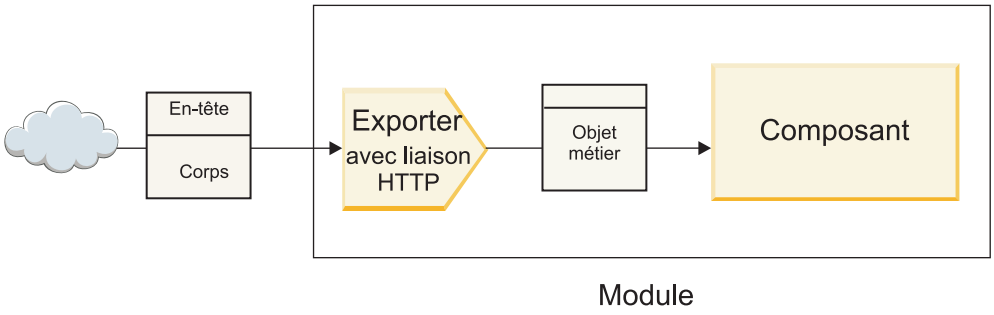


Figure 46. Exportation avec une liaison HTTP

Importations

Il est possible qu'un composant SCA souhaite appeler un service externe non SCA qui attend des données dans un format différent. Une importation est utilisée par le composant SCA pour appeler le service externe à l'aide du modèle de programmation SCA. L'importation appelle ensuite le service cible selon les attentes du service.

Par exemple, dans la figure suivante, une requête provenant d'un composant SCA est envoyée par l'importation vers un service externe. L'objet métier qui est au format utilisé par le composant SCA est transformé en format attendu par le service et le service est appelé.

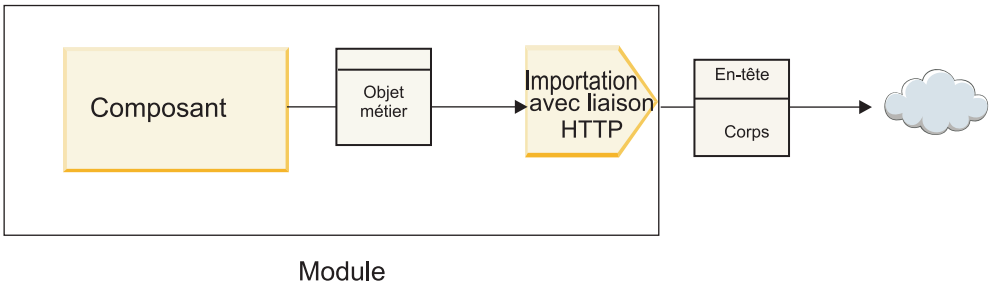


Figure 47. Importation avec une liaison HTTP

Liste de liaisons

Integration Designer permet de générer une liaison pour une importation ou une exportation et de configurer la liaison. Les types de liaisons disponibles sont décrites dans la liste suivante.

- SCA

  La liaison SCA qui est la liaison par défaut permet à votre service de communiquer avec des services dans d'autres modules SCA. Une importation avec une liaison SCA permet d'accéder à un service dans un autre module SCA. Une exportation avec une liaison SCA permet d'offrir un service à d'autres modules SCA.
- Service Web

  Une liaison de service Web vous permet d'accéder à un service externe par le biais de messages SOAP interopérables et des qualités de service. Vous pouvez également utiliser des liaisons de service Web pour inclure des pièces jointes dans le message SOAP.

  La liaison de service Web peut utiliser un protocole de transport de type SOAP/HTTP (SOAP via HTTP) ou SOAP/JMS (SOAP via JMS). Quel que soit le transport (HTTP ou JMS) utilisé pour transmettre les messages SOAP, les liaisons de service Web traitent toujours les interactions de demande/réponse de manière synchrone.

- HTTP

  La liaison HTTP vous permet d'accéder à un service externe via le protocole HTTP lorsque des messages non SOAP sont utilisés ou lorsque l'accès HTTP direct est requis. Cette liaison est utilisée lorsque vous utilisez des services Web qui sont basés sur le modèle HTTP (c'est-à-dire les services qui utilisent des opérations d'interface HTTP bien connues comme GET, PUT, DELETE, etc.).
- EJB (Enterprise JavaBeans)

  Les liaisons EJB permettent aux composants SCA d'interagir avec les services fournis par la logique métier Java EE exécutée sur un serveur Java EE.
- EIS

  Lorsqu'elle est utilisée avec un adaptateur de ressources JCA, la liaison EIS (Enterprise Information System) permet d'accéder à des services sur un système d'information d'entreprise ou de rendre vos services disponibles auprès de l'EIS.
- Liaisons JMS

  Les liaisons Java Message Service (JMS), JMS génériques et JMS WebSphere MQ (JMS MQ) sont utilisées pour des interactions avec des systèmes de messagerie où la communication asynchrone via les files d'attente de messages est critique pour la fiabilité.

  Une exportation avec une des liaisons JMS surveille l'arrivée d'un message dans une file d'attente et la réponse, le cas échéant, est envoyée de manière asynchrone à la file d'attente de réponses. Une importation avec une des liaisons JMS génère et envoie un message vers une file d'attente JMS et surveille, le cas échéant, l'arrivée d'une réponse dans une file d'attente.
  - JMS

    La liaison JMS vous permet d'accéder au fournisseur JMS imbriqué dans WebSphere.
  - JMS générique

    La liaison JMS générique vous permet d'accéder à un système de messagerie fournisseur non IBM.
  - JMS MQ

    La liaison JMS MQ vous permet d'accéder au sous-ensemble JMS d'un système de messagerie WebSphere MQ. Vous pourriez utiliser cette liaison lorsque le sous-ensemble JMS de fonctions est suffisant pour votre application.
- MQ

  La liaison WebSphere MQ vous permet de communiquer avec des applications natives MQ, en les insérant dans le cadre de l'architecture SOA et en fournissant un accès aux informations d'en-tête propres à MQ. Cette liaison est utile lorsque vous devez utiliser des fonctions natives MQ.

Présentation des liaisons d'importation et d'exportation

Une exportation vous permet de rendre les services d'un module d'intégration disponibles pour les clients externes et une importation permet à vos composants SCA au sein d'un module d'intégration d'appeler des services externes. La liaison associée à l'exportation ou à l'importation spécifie la relation entre les messages de protocole et les objets métier. Elle spécifie également la manière dont les opérations et les erreurs sont sélectionnées.

Flux d'informations via une exportation

Une exportation reçoit une requête destinée au composant auquel l'exportation est connectée via un transport spécifique déterminé par la liaison associée (par exemple, HTTP).

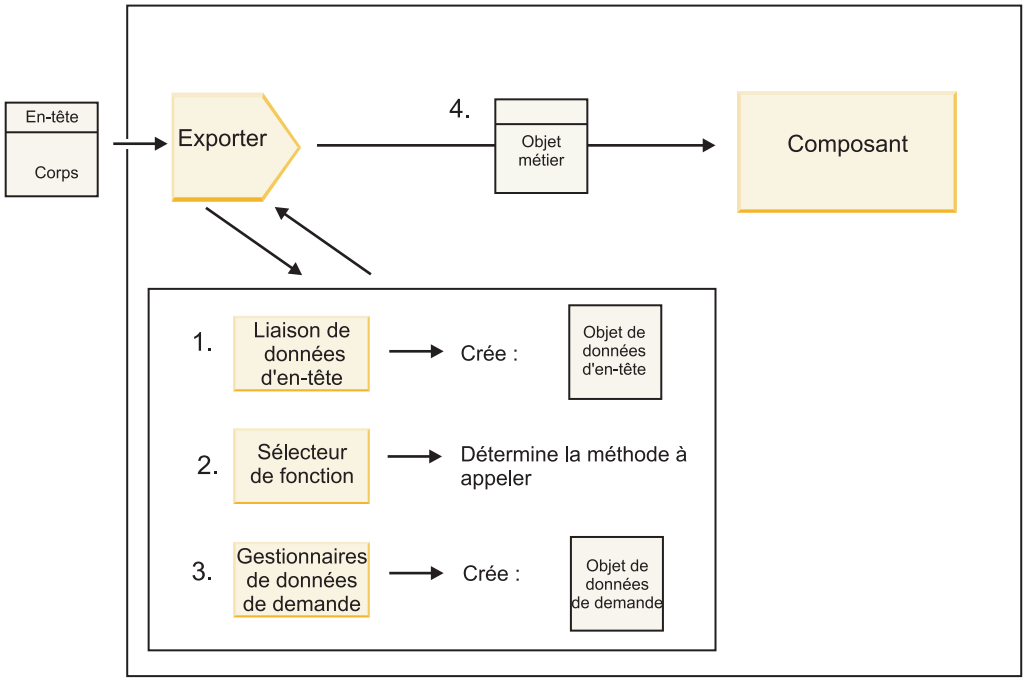


Figure 48. Flux d'une requête via l'exportation vers un composant

Lorsque l'exportation reçoit la requête, la séquence d'événements suivante se produit :

1. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête transforme l'en-tête de protocole en un objet de données d'en-tête.
2. Le sélecteur de fonction détermine le nom de la méthode native à partir du message de protocole. Le nom de la méthode native est mappée par la configuration d'exportation vers le nom d'une opération sur l'interface de l'exportation.
3. Le gestionnaire de données de requête ou la liaison de données sur la méthode transforme la requête en un objet métier de type requête.
4. L'exportation appelle la méthode de composant avec l'objet métier de type requête.
   - La liaison d'exportation HTTP, la liaison d'exportation des services Web, et la liaison d'exportation EJB appellent le composant SCA de manière synchrone.
   - Les liaisons d'exportation JMS, Generic JMS, MQ JMS et WebSphere MQ appellent le composant SCA de manière asynchrone.

Notez qu'une exportation peut propager les propriétés d'en-tête et utilisateur qu'elle reçoit via le protocole , si la propagation de contexte est activée. Les composants qui sont reliés à l'exportation peuvent ensuite accéder à ces propriétés d'en-têtes et utilisateur. Pour plus d'informations, consultez la rubrique «Propagation» du centre de documentation de WebSphere Integration Developer.

S'il s'agit d'une opération bidirectionnelle, le composant renvoie une réponse.

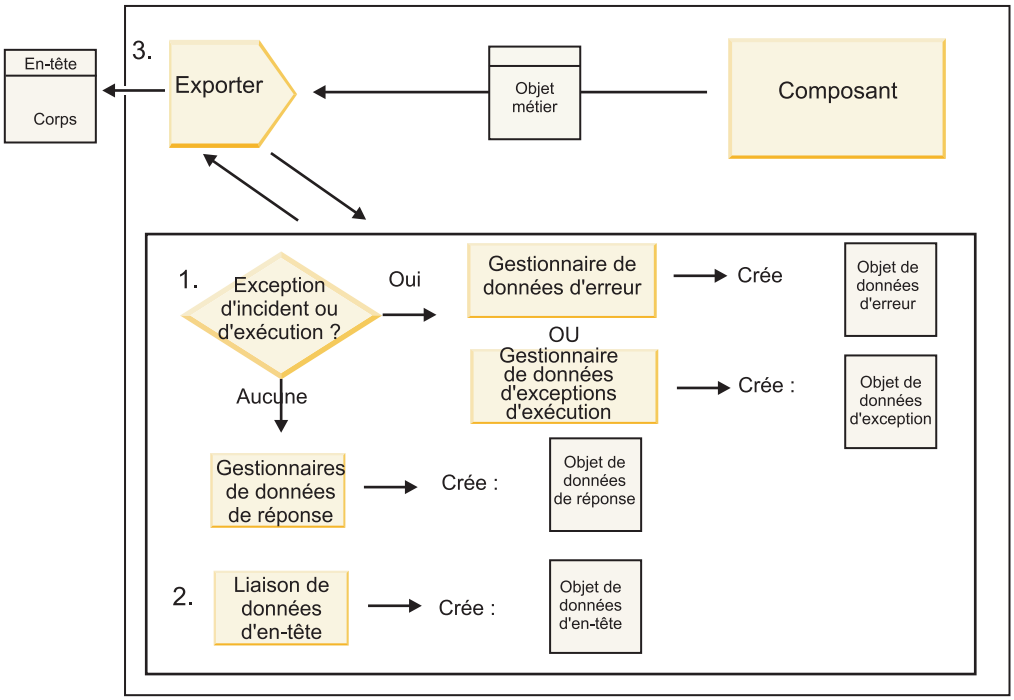


Figure 49. Flux d'une réponse en retour via l'exportation

La séquence d'étapes suivante se produit :

1. Si un message de réponse normal est reçu par la liaison d'exportation, le gestionnaire des données de réponse ou la liaison de données sur la méthode transforme l'objet métier en réponse.

   Si la réponse est une erreur, le gestionnaire des données d'erreur ou la liaison de données sur la méthode transforme l'erreur en une réponse d'erreur.

   Pour les liaisons d'exportation HTTP uniquement, si la réponse est une exception d'exécution, le gestionnaire des données d'exception d'exécution est appelé s'il est configuré.
2. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête transforme les objets de données d'en-tête en en-têtes de protocole.
3. L'exportation envoie la réponse du service via le transport.

Flux d'informations via une importation

Les composants envoient des requêtes aux services en dehors du module en utilisant une importation. La requête est envoyée via un transport spécifique déterminé par la liaison associée.

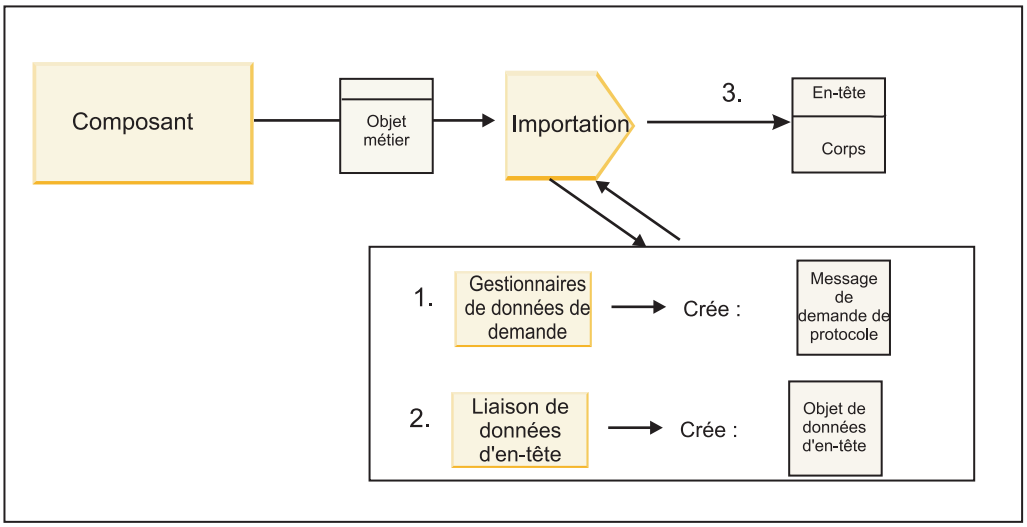


Figure 50. Flux d'un composant vers un service via l'importation

Le composant appelle l'importation avec un objet métier de type requête.

Remarque :

- La liaison d'importation HTTP, la liaison d'importation du service Web et la liaison d'importation EJB doivent être appelés de façon synchrone par le composant appelant.
- Les liaisons d'importation JMS, Generic JMS, MQ JMS et WebSphere MQ doivent être appelées de manière asynchrone.

Une fois que le composant a appelé l'importation, la séquence d'événements suivante se produit :

1. Le gestionnaire de données de requête ou la liaison de données sur la méthode transforme l'objet métier de type requête en un message de requête de protocole.
2. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête sur la méthode définit l'objet métier d'en-tête dans l'en-tête de protocole.
3. L'importation appelle le service avec la requête de service via le transport.

S'il s'agit d'une opération bidirectionnelle, le service renvoie une réponse et la séquence d'événements suivante se produit :

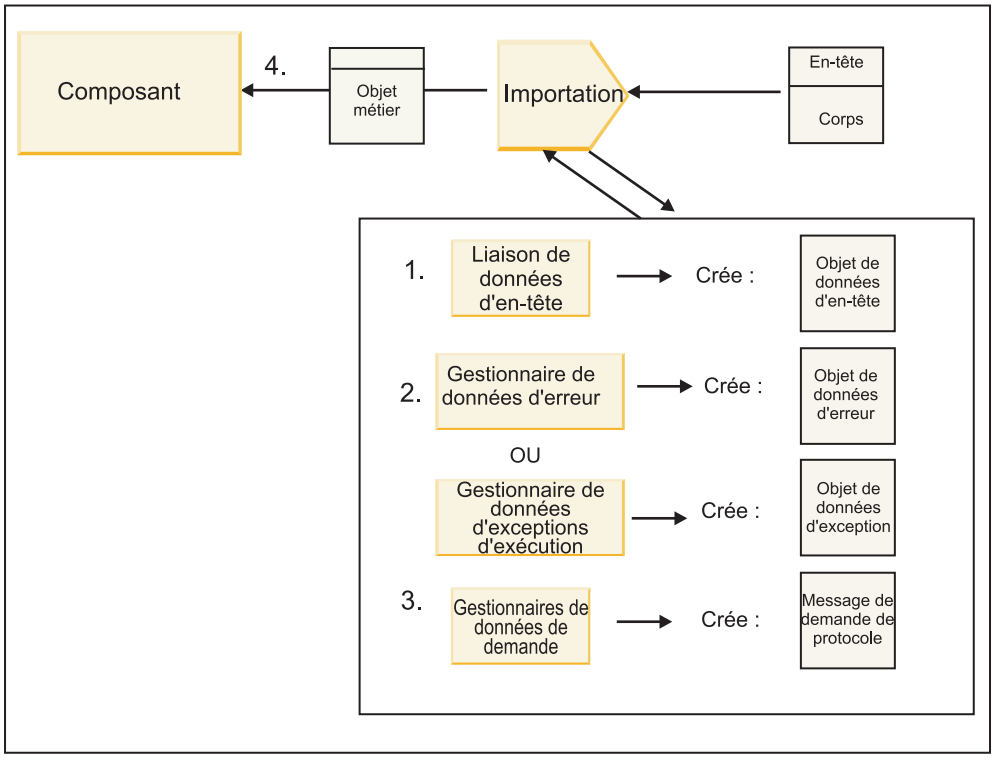


Figure 51. Flux d'une réponse en retour via l'importation

1. Pour les liaisons WebSphere MQ uniquement, la liaison des données d'en-tête transforme l'en-tête de protocole en un objet de données d'en-tête.
2. Identification de la réponse afin de déterminer s'il s'agit d'une erreur.
   - Si la réponse est une erreur, le sélecteur d'erreurs inspecte l'erreur afin de déterminer vers quelle erreur WSDL elle mappe. Le gestionnaire des données d'erreur sur la méthode transforme ensuite l'erreur en une réponse d'erreur.
   - Si la réponse est une exception d'exécution, le gestionnaire des données d'exception d'exécution est appelé s'il est configuré.
3. Le gestionnaire de données de réponse ou la liaison sur la méthode transforme la réponse en un objet métier de réponse.
4. L'importation renvoie l'objet métier de réponse au composant.

Configuration des liaisons d'importation et d'exportation

Un des aspects clés des liaisons d'importation et d'exportation est constitué par la transformation du format des données qui indique la manière dont les données sont mappées (désérialisées) à partir d'un format de connexion natif vers un objet métier ou la manière dont elles sont mappées (sérialisées) depuis un objet métier vers un format de connexion natif. Pour les liaisons associées aux exportations, vous pouvez également spécifier un sélecteur de fonction pour identifier les opérations qui doivent être effectuées sur les données. Pour les liaisons associées aux exportations ou aux importations, vous pouvez indiquer le mode de gestion des échecs qui se sont produits lors du traitement.

En outre, vous pouvez spécifier des informations spécifiques au transport sur les liaisons. Par exemple, pour une liaison HTTP, vous pouvez spécifier l'adresse URL de noeud final. Pour la liaison HTTP, les informations spécifiques au transport sont décrites dans les rubriques «Génération d'une liaison d'importation HTTP» et «Génération d'une liaison d'exportation HTTP». Vous pouvez également trouver des informations sur les autres liaisons dans le centre de documentation.

Transformation du format des données dans les importations et exportations

Lorsqu'une liaison d'exportation ou d'importation est configurée dans IBM Integration Designer, une des propriétés de configuration spécifiées concerne le format de données utilisé par la liaison.

- Pour les liaisons d'exportation, dans lesquelles une application client envoie des requêtes à un composant SCA et reçoit des réponses en retour, vous indiquez le format des données natives. En fonction du format, le système sélectionne le gestionnaire de données approprié ou la liaison de données pour transformer les données natives en un objet métier (qui est utilisé par le composant SCA) et inversement pour transformer l'objet métier en données natives (qui est la réponse à l'application client).
- Pour les liaisons d'importation, dans lesquelles un composant SCA envoie des requêtes à un service en dehors du module et reçoit des réponses en retour, vous indiquez le format de données des données natives. En fonction du format, le système sélectionne le gestionnaire de données approprié ou la liaison de données pour transformer l'objet métier en données natives et inversement.

IBM Business Process Manager fournit un ensemble de formats de données prédéfinis et de gestionnaires de données correspondants ou de liaisons de données qui prennent en charge les formats. Vous pouvez également créer vos propres gestionnaires de données personnalisés et enregistrer le format de données pour ces gestionnaires de données. Pour plus d'informations, voir «Développement des gestionnaires de données» dans le centre de documentation de IBM Integration Designer.

- Les *gestionnaires de données* ne dépendent pas d'un protocole et peuvent transformer les données d'un format à un autre. Dans IBM Business Process Manager, les gestionnaires de données transforment généralement des données natives (comme XML, CSV et COBOL) en un objet métier et inversement. Puisqu'ils ne dépendent pas d'un protocole, vous pouvez réutiliser le même gestionnaire de données avec toute une variété de liaisons d'exportation et d'importation. Par exemple, vous pouvez utiliser le même gestionnaire de données XML avec une liaison d'exportation ou d'importation HTTP ou avec une liaison d'exportation ou d'importation JMS.
- Les *liaisons de données* transforment également des données natives en objet métier (et inversement), mais elles sont spécifiques à un protocole. Par exemple, une liaison de données HTTP peut être utilisée uniquement avec une liaison d'exportation ou d'importation HTTP. Contrairement aux gestionnaires de données, une liaison de données HTTP ne peut pas être réutilisée avec une liaison d'exportation ou d'importation MQ.

  Remarque : Trois liaisons de données HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML et HTTPServiceGatewayDataBinding) sont dépréciées à partir de IBM Business Process Manager, version 7.0. Utilisez des gestionnaires de données chaque fois que possible.

Comme indiqué précédemment, vous pouvez créer des gestionnaires de données personnalisés, si nécessaire. Vous pouvez également créer des liaisons de données personnalisées. Toutefois, il est recommandé de créer des gestionnaires de données personnalisés car ils peuvent être utilisés avec plusieurs liaisons.

Gestionnaires de données :

Les gestionnaires de données sont configurés en fonction des liaisons d'importation et d'exportation pour transformer les données d'un format à un autre dans un environnement de protocole neutre. Plusieurs gestionnaires de données sont fournis avec le produit, mais vous pouvez également créer votre propre gestionnaire de données si nécessaire. Vous pouvez associer un gestionnaire de données à une liaison d'importation ou d'exportation à l'un des deux niveaux disponibles : vous pouvez l'associer à toutes les opérations dans l'interface de l'importation ou de l'exportation ou bien vous pouvez l'associer à une opération spécifique pour la requête ou la réponse.

Gestionnaires de données prédéfinis

IBM Integration Designer permet de spécifier le gestionnaire de données que vous voulez utiliser.

Le tableau suivant répertorie les gestionnaires de données prédéfinis et décrit également la manière dont chaque gestionnaire de données transforme les données entrantes et sortantes.

Remarque : Sauf indication contraire, ces gestionnaires de données peuvent être utilisés avec des liaisons JMS, Generic JMS, JMS MQ, WebSphere MQ et HTTP.
Voir la rubrique «Gestionnaires de données» dans le centre de documentation de Integration Designer pour obtenir des informations détaillées.

Tableau 52. Gestionnaires de données prédéfinis

Gestionnaire de données	Données natives vers objet métier	Objet métier vers données natives
ATOM	Analyse les flux ATOM dans un objet métier de flux ATOM.	Sérialise un objet métier de flux ATOM vers des flux ATOM.
Délimité	Analyse les données délimitées dans un objet métier.	Sérialise un objet métier vers des données délimitées, y compris CSV.
Largeur fixe	Analyse les données à largeur fixe dans un objet métier.	Sérialise un objet métier vers des données à largeur fixe.
Géré par WTX	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est dérivé par le gestionnaire de données.	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est dérivé par le gestionnaire de données.
Géré par le générateur d'appels WTX	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est fourni par l'utilisateur.	Délègue la transformation du format des données à WebSphere Transformation Extender (WTX). Le nom de mappe WTX est fourni par l'utilisateur.
JAXB	Sérialise des beans Java sous la forme d'objet métier à l'aide de règles de mappage définies par la spécification JAXB (Java Architecture for XML Binding).	Désérialise un objet métier Java à l'aide des règles de mappage définies par la spécification JAXB.
JAXWS **Remarque :** Le gestionnaire de données JAXWS ne peut être utilisé qu'avec la liaison EJB.	Utilisé par une liaison d'EJB pour transformer un objet Java de réponse ou un objet Java d'exception en objet métier de réponse à l'aide des règles de mappage définies par la spécification JAX-WS (Java API for XML Web Services).	Utilisé par une liaison d'EJB pour transformer un objet métier en paramètre de méthode Java à l'aide des règles de mappage définies par la spécification JAX-WS.
JSON	Analyse les données JSON dans un objet métier.	Sérialise un objet métier vers des données JSON.
Corps natif	Effectue l'analyse syntaxique des octets natifs (texte, mappe, flux ou objet) pour obtenir un des cinq objets métier de base (texte, octets, mappe, flux ou objet).	Transforme les cinq objets métier de base en octets, texte, mappe, flux ou objet.
SOAP	Analyse le message SOAP (et l'en-tête) dans un objet métier.	Sérialise un objet métier vers un message SOAP.
XML	Analyse les données XML dans un objet métier.	Sérialise un objet métier vers des données XML.
UTF8XMLDataHandler	Analyse les données XML codées en UTF-8 dans un objet métier.	Sérialise un objet métier dans des données XML codées en UTF-8 lors de l'envoi d'un message.

Création d'un gestionnaire de données

Pour plus d'informations sur la création d'un gestionnaire de données, consultez la rubrique «Développement des gestionnaires de données» du centre de documentation de Integration Designer.

Liaisons de données :

Les liaisons de données sont configurées en fonction des liaisons d'importation et d'exportation pour transformer les données d'un format à un autre. Les liaisons de données sont spécifiques à un protocole. Plusieurs liaisons de données sont fournies avec le produit, mais vous pouvez également créer votre propre liaison de données si nécessaire. Vous pouvez associer une liaison de données à une liaison d'importation ou d'exportation sur l'un des deux niveaux disponibles – vous pouvez l'associer avec toutes les opérations dans l'interface de l'importation ou de l'exportation ou bien vous pouvez l'associer à une opération spécifique pour la requête ou la réponse.

IBM Integration Designer vous permet de spécifier quelle liaison de données vous voulez utiliser ou de créer votre propre liaison de données. Une discussion traitant des liaisons de données est disponible dans la section «Présentation des liaisons JMS, JMS MQ et JMS génériques» du centre de documentation de IBM Integration Designer.

Liaisons JMS

Le tableau suivant répertorie les liaisons de données qui peuvent être utilisées :

- Liaisons JMS
- Liaisons JMS génériques
- Liaisons JMS WebSphere MQ

Le tableau comprend également une description des tâches que les liaisons de données effectuent.

Tableau 53. Liaisons de données prédéfinies pour les liaisons JMS

Liaison de données	Données natives vers objet métier	Objet métier vers données natives
Objet Java sérialisé	Transforme l'objet sérialisé Java en un objet métier (qui est mappé en tant que type d'entrée ou de sortie dans WSDL).	Sérialise un objet métier vers un objet sérialisé Java dans le message d'objet JMS.
Octets encapsulés	Extrait les octets du message d'octets JMS entrant et les encapsule dans l'objet métier JMSBytesBody.	Extrait les octets de l'objet métier JMSBytesBody et les encapsule dans le message d'octets JMS sortant
Entrée de mappe encapsulée	Extrait les informations de nom, de valeur et de type pour chaque entrée dans le message de mappe JMS entrant et crée une liste d'objets métier MapEntry. Encapsule ensuite la liste dans l'objet métier JMSMapBody	Extrait les informations de nom, de valeur et de type à partir de la liste MapEntry dans l'objet métier JMSMapBody business et crée les entrées correspondantes dans le message de mappe JMS sortant.
Objet encapsulé	Extrait l'objet du message d'objet JMS entrant et l'encapsule dans l'objet métier JMSObjectBody.	Extrait l'objet de l'objet métier JMSObjectBody et l'encapsule dans le message d'objet JMS sortant.
Texte encapsulé	Extrait le texte du message de texte JMS entrant et l'encapsule dans l'objet métier JMSTextBody.	Extrait le texte de l'objet métier JMSTextBody et l'encapsule dans le message de texte JMS sortant.

Liaisons WebSphere MQ

Le tableau suivant répertorie les liaisons de données qui peuvent être utilisées avec WebSphere MQ et décrit les tâches que les liaisons de données peuvent effectuer.

Tableau 54. Liaisons de données prédéfinies pour les liaisons WebSphere MQ

Liaison de données	Données natives vers objet métier	Objet métier vers données natives
Objet Java sérialisé	Transforme l'objet sérialisé Java à partir du message entrant en un objet métier (qui est mappé en tant que type d'entrée ou de sortie dans WSDL).	Transforme un objet métier en objet sérialisé Java dans le message sortant
Octets encapsulés	Extrait les octets du message d'octets MQ non structuré et les encapsule dans l'objet métier JMSBytesBody.	Extrait les octets d'un objet métier JMSBytesBody et encapsule ces octets dans le message d'octets MQ non structuré sortant.
Texte encapsulé	Extrait le texte du message de texte MQ non structuré et l'encapsule dans un objet métier JMSTextBody.	Extrait le texte d'un objet métier JMSTextBody et l'encapsule dans un message de texte MQ non structuré.
Entrée de flux encapsulée	Extrait les informations de nom et de type pour chaque entrée dans le message de flux JMS entrant et crée une liste d'objets métier StreamEntry. Encapsule ensuite la liste dans l'objet métier JMSStreamBody.	Extrait les informations de nom et de type à partir de la liste StreamEntry dans l'objet métier JMSStreamBody et crée les entrées correspondantes dans le message JMSStreamMessage sortant.

Outre les liaisons de données répertoriées dans le tableau 25, à la page 61, WebSphere MQ utilise également des liaisons de données d'en-tête. Pour plus d'informations, reportez-vous au centre de documentation de IBM Integration Designer.

Liaisons HTTP

Le tableau suivant répertorie les liaisons de données qui peuvent être utilisées avec HTTP et décrit les tâches que les liaisons de données peuvent effectuer.

Tableau 55. Liaisons de données prédéfinies pour les liaisons HTTP

Liaison de données	Données natives vers objet métier	Objet métier vers données natives
Octets encapsulés	Extrait les octets du corps du message HTTP entrant et les encapsule dans l'objet métier HTTPBytes.	Extrait les octets de l'objet métier HTTPBytes et les ajoute au corps du message HTTP sortant.
Texte encapsulé	Extrait le texte du corps du message HTTP entrant et l'encapsule dans l'objet métier HTTPText.	Extrait le texte de l'objet métier HTTPText et l'ajoute au corps du message HTTP sortant.

Sélecteurs de fonction dans les liaisons d'exportation

Un sélecteur de fonction permet d'identifier les opérations qui doivent être effectuées sur les données pour un message de demande. Les sélecteurs de fonction sont configurés dans le cadre d'une liaison d'exportation.

Considérons une exportation SCA qui expose une interface. L'interface contient deux opérations : Créer et Mettre à jour. L'exportation a une liaison JMS qui lit à partir d'une file d'attente.

Lorsqu'un message arrive dans la file d'attente, l'exportation est transmise aux données associées, mais quelle opération de l'interface d'exportation doit être appelée sur le composant connecté ? L'opération est déterminée par le sélecteur de fonction et la configuration de la liaison d'exportation.

Le sélecteur de fonction renvoie le nom de la fonction native (le nom de la fonction dans le système client ayant envoyé le message). Le nom de la fonction native est ensuite mappé à l'opération ou au nom de la fonction sur l'interface associée à l'exportation. Par exemple, dans la figure suivante, le sélecteur de fonction renvoie le nom de la fonction native (CRT) à partir du message entrant, le nom de la fonction native est mappé à l'opération Créer et l'objet métier est envoyé au composant SCA avec l'opération Créer.

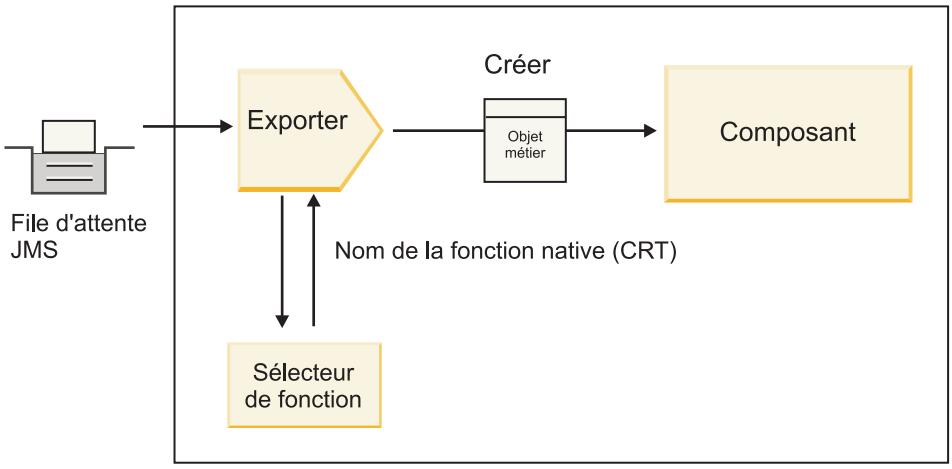


Figure 52. Sélecteur de fonction

Si l'interface ne présente qu'une opération, il n'est pas nécessaire de spécifier un sélecteur de fonction.

Plu sieurs sélecteurs de fonction préintégrés sont disponibles et répertoriés dans les sections ci-dessous.

Liaisons JMS

Le tableau suivant répertorie les sélecteurs de fonction qui peuvent être utilisés avec :

- Liaisons JMS
- Liaisons JMS génériques
- Liaisons JMS WebSphere MQ

Tableau 56. Sélecteurs de fonction prédéfinis pour les liaisons JMS

Sélecteur de fonction	Description
Sélecteur de fonction JMS pour des liaisons de données JMS simples	Utilise la propriété JMSType du message pour sélectionner le nom de l'opération.
Sélecteur de fonction de propriété d'en-tête JMS	Renvoie la valeur de la propriété de chaîne JMS, TargetFunctionName, à partir de l'en-tête.
Sélecteur de fonction de passerelle de service JMS	Détermine si la requête est une opération unidirectionnelle ou bidirectionnelle en examinant l'ensemble de propriétés JMSReplyTo par le client.

Liaisons WebSphere MQ

Le tableau suivant répertorie les sélecteurs de fonction qui peuvent être utilisés avec les liaisonsWebSphere MQ.

Tableau 57. Sélecteurs de fonction prédéfinis pour les liaisons WebSphere MQ

Sélecteur de fonction	Description
Sélecteur de fonction MQ handleMessage	Renvoie un message handleMessage comme valeur qui est mappé à l'aide des liaisons de méthode d'exportation au nom d'une opération sur l'interface.
MQ utilise le sélecteur de fonction JMS par défaut	Lit l'opération native à partir de la propriété TargetFunctionName du dossier d'un en-tête MQRFH2.
MQ utilise le format du corps de message comme fonction native	Recherche la zone Format du dernier en-tête et renvoie cette zone sous forme de chaîne.
Sélecteur de fonction de type MQ	Crée une méthode dans votre liaison d'exportation par extraction de l'adresse URL contenant les propriétés Msd, Set, Type et Format de l'en-tête MQRFH2.
Sélecteur de fonction de passerelle de service MQ	Utilise la propriété MsgType de l'en-tête MQMD pour déterminer le nom de l'opération.

Liaisons HTTP

Le tableau suivant répertorie les sélecteurs de fonction qui peuvent être utilisés avec les liaisons HTTP.

Tableau 58. Sélecteurs de fonction prédéfinis pour les liaisons HTTP

Sélecteur de fonction	Description
Sélecteur de fonction HTTP basé sur l'en-tête TargetFunctionName	Utilise la propriété de l'en-tête HTTP TargetFunctionName du client pour déterminer l'opération à appeler au moment de l'exécution à partir de l'exportation.
Sélecteur de fonction HTTP basé sur l'adresse URL et la méthode HTTP	Utilise le chemin relatif à partir de l'adresse URL ajoutée avec la méthode HTTP à partir du client pour déterminer l'opération native définie sur l'exportation.
Sélecteur de fonction de passerelle de service HTTP basé sur l'adresse URL avec un nom d'opération	Détermine la méthode à appeler en fonction de l'adresse URL si "operationMode = oneway" a été ajouté à l'URL de demande.

Remarque : Vous pouvez également créer votre propre sélecteur de fonction à l'aide d'IBM Integration Designer. Les informations de création d'un sélecteur de fonction se trouvent dans le centre de documentation d'IBM Integration Designer. Par exemple, une description de la création d'un sélecteur de fonction pour les liaisons WebSphere MQ est disponible dans la rubrique «Présentation des sélecteurs de fonction MQ».

Gestion des erreurs

Vous pouvez configurer vos liaisons d'importation et d'exportation pour gérer les erreurs (par exemple, les exceptions métier) qui se produisent lors du traitement en spécifiant des gestionnaires de données d'erreur. Vous pouvez configurer un gestionnaire de données d'erreur sur trois niveaux : vous pouvez associer un gestionnaire de données d'erreur à une erreur, à une opération ou pour toutes les opérations à une liaison.

Un gestionnaire de données d'erreur traite les données d'erreur et les convertit au format correct pour les envoyer via la liaison d'importation ou d'exportation.

- Pour une liaison d'exportation, le gestionnaire de données d'erreur transforme l'objet métier d'exception envoyé depuis le composant en un message de réponse qui peut être utilisé par l'application client.
- Pour une liaison d'importation, le gestionnaire de données d'erreur transforme les données d'erreur ou le message de réponse provenant d'un service en un objet métier d'exception qui peut être utilisé par le composant SCA.

Pour les liaisons d'importation, la liaison appelle le sélecteur d'erreurs qui détermine si le message de réponse est une réponse normale, une erreur métier ou une exception d'exécution.

Vous pouvez spécifier un gestionnaire de données d'erreur pour une erreur, une opération particulière et pour toutes les opérations avec une liaison.

- Si le gestionnaire de données d'erreur est défini sur les trois niveaux, le gestionnaire de données associé à une erreur particulière est appelé.
- Si les gestionnaires de données d'erreur sont définis aux niveaux de l'opération et de la liaison, le gestionnaire de données associé à l'opération est appelé.

Deux éditeurs sont utilisés dans IBM Integration Designer pour spécifier la gestion des erreurs. L'éditeur d'interface est utilisé pour indiquer s'il y aura une erreur sur une opération. Après avoir généré une liaison avec cette interface, l'éditeur de la vue des propriétés vous permet de configurer la manière dont l'erreur sera gérée. Pour plus d'informations, voir la rubrique «Sélecteur d'erreur» dans le centre de documentation de IBM Integration Designer.

Gestion des erreurs dans les liaisons d'exportation :

Lorsqu'une erreur se produit lors du traitement de la requête d'une application client, la liaison d'exportation peut renvoyer les informations d'erreur au client. Vous pouvez configurer la liaison d'exportation pour spécifier la manière dont l'erreur doit être traitée et renvoyée au client.

Vous configurez la liaison d'exportation à l'aide de IBM Integration Designer.

Lors du traitement de la requête, un client appelle une exportation avec une requête et l'exportation appelle le composant SCA. Lors du traitement de la requête, le composant SCA peut renvoyer une réponse métier ou émettre une exception métier de service ou une exception d'exécution de service. Lorsque cela se produit, la liaison d'exportation transforme l'exception en un message d'erreur et l'envoie au client, comme indiqué dans la figure ci-dessous et décrit dans les sections suivantes.

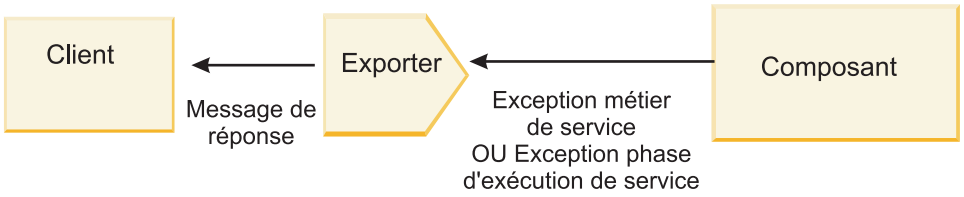


Figure 53. Manière dont les informations sont envoyées du composant vers le client via la liaison d'exportation

Vous pouvez créer un gestionnaire de données personnalisé ou une liaison de données pour gérer les erreurs.

Erreurs métier

Les erreurs métier sont des erreurs ou des exceptions métier qui se produisent lors du traitement.

Considérez l'interface suivante présentant une opération createCustomer. Cette opération se caractérise par deux erreurs métier définies : CustomerAlreadyExists et MissingCustomerId.

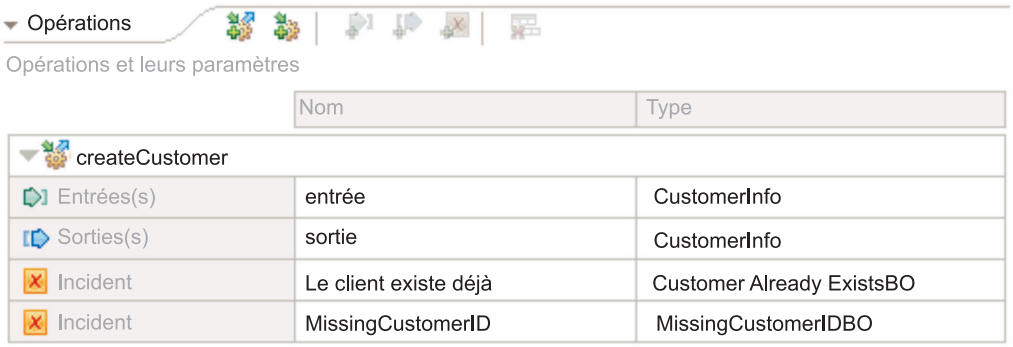


Figure 54. Interface avec deux erreurs

Dans cet exemple, si un client envoie une requête pour créer un client (à ce composant SCA) et que le client existe déjà, le composant émet une erreur CustomerAlreadyExists vers l'exportation. L'exportation doit propager cette erreur métier en retour au client appelant. Pour ce faire, elle utilise le gestionnaire de données configuré sur la liaison d'exportation.

Lorsqu'une erreur métier est reçue par la liaison d'exportation, le traitement suivant se produit :

1. La liaison détermine le gestionnaire de données d'erreurs à appeler pour gérer les erreurs. Si l'exception métier de service contient le nom de l'erreur, le gestionnaire de données configuré sur l'erreur est appelé. Si l'exception métier de service ne contient pas le nom de l'erreur, ce nom est dérivé en faisant correspondre les types d'erreur.
2. La liaison appelle le gestionnaire de données d'erreurs avec l'objet de données à partir de l'exception métier de service.
3. Le gestionnaire de données d'erreur transforme l'objet données d'erreur en un message de réponse et le renvoie à la liaison d'exportation.
4. L'exportation renvoie le message de réponse au client.

Si l'exception métier de service contient le nom de l'erreur, le gestionnaire de données configuré sur l'erreur est appelé. Si l'exception métier de service ne contient pas le nom de l'erreur, ce nom est dérivé en faisant correspondre les types d'erreur.

Exceptions d'exécution

Une exception d'exécution est une exception qui se produit dans l'application SCA lors du traitement d'une requête qui ne correspond pas à une erreur métier. Contrairement aux erreurs métier, les exceptions d'exécution ne sont pas définies sur l'interface.

Dans certains scénarios, il est possible que vous souhaitiez propager ces exceptions d'exécution à l'application client de telle sorte que l'application client puisse entreprendre l'action appropriée.

Par exemple, si un client envoie une requête (au composant SCA) pour créer un client et qu'une erreur d'autorisation se produit lors du traitement de la requête, le composant émet une exception d'exécution. Cette exception d'exécution doit être propagée en retour au client appelant de telle sorte qu'il puisse entreprendre l'action appropriée en ce qui concerne l'autorisation. Cette opération est accomplie par le gestionnaire de données d'exception d'exécution configuré sur la liaison d'exportation.

Remarque : Vous pouvez configurer un gestionnaire de données d'exception d'exécution uniquement sur des liaisons HTTP.

Le traitement d'une exception d'exécution est identique au traitement d'une erreur métier. Si un gestionnaire de données d'exception d'exécution était configuré, le traitement suivant se produit :

1. La liaison d'exportation appelle le gestionnaire de données approprié avec l'exception d'exécution de service.
2. Le gestionnaire de données transforme l'objet données d'erreur en un message de réponse et le renvoie à la liaison d'exportation.
3. L'exportation renvoie le message de réponse au client.

La gestion des erreurs et la gestion des exceptions d'exécution sont facultatives. Si vous ne voulez pas propager les erreurs et les exceptions d'exécution au client appelant, ne configurez pas le gestionnaire de données d'erreur ou le gestionnaire de données d'exception d'exécution.

Gestion des erreurs dans les liaisons d'importation :

Un composant utilise une importation pour envoyer une requête à un service en dehors du module. Lorsqu'une erreur se produit lors du traitement de la requête, le service renvoie l'erreur à la liaison d'importation. Vous pouvez configurer la liaison d'importation pour spécifier la manière dont l'erreur doit être traitée et renvoyée au composant.

Vous configurez la liaison d'importation à l'aide de IBM Integration Designer. Vous pouvez spécifier un gestionnaire de données d'erreur (ou une liaison de données) et vous pouvez également spécifier un sélecteur d'erreurs.

Gestionnaires de données d'erreur

Le service qui traite la requête envoie des informations d'erreur à la liaison d'importation sous la forme d'une exception ou un message de réponse contenant les données d'erreur.

La liaison d'importation transforme l'exception de service ou le message de réponse en une exception métier de service ou une exception d'exécution de service, comme indiqué dans la figure ci-dessous et décrit dans les sections suivantes.

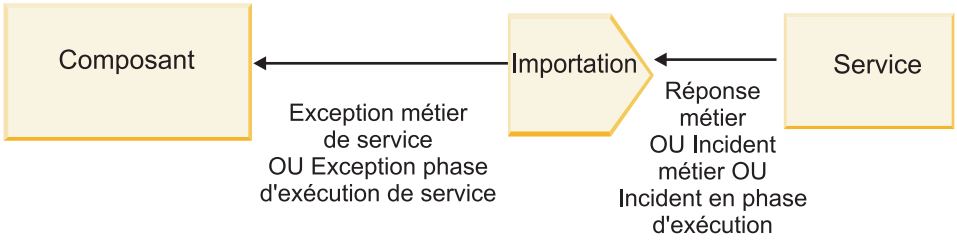


Figure 55. Manière dont les informations d'erreur sont envoyées depuis le service au composant via l'importation

Vous pouvez créer un gestionnaire de données personnalisé ou une liaison de données pour gérer les erreurs.

Sélecteurs d'erreurs

Lorsque vous configurez une liaison d'importation, vous pouvez spécifier un sélecteur d'erreurs. Le sélecteur d'erreurs détermine si la réponse d'importation est une réponse réelle, une exception métier ou une erreur d'exécution. Il détermine également, à partir de l'en-tête ou du corps de la réponse, le nom de l'erreur native qui est mappé par la configuration de liaison au nom d'une erreur dans l'interface associée.

Deux types de sélecteurs d'erreurs préintégrés sont disponibles avec les importations JMS, JMS MQ, JMS génériques, WebSphere MQ et HTTP :

Tableau 59. Sélecteurs d'erreurs préintégrés

Type de sélecteur d'erreurs	Description
Basé sur les en-têtes	Indique si un message de réponse est une erreur métier, une exception d'exécution ou un message normal en fonction des en-têtes dans le message de réponse.
SOAP	Détermine si le message de réponse SOAP est une réponse normale, une erreur métier ou une exception d'exécution.

Les exemples ci-dessous illustrent des sélecteurs d'erreurs basés sur les en-têtes et le sélecteur d'erreurs SOAP.

- Sélecteur d'erreurs basé sur les en-têtes

  Si une application veut indiquer que le message entrant est une erreur métier, il doit y avoir deux en-têtes dans le message entrant pour les erreurs métier, comme indiqué ci-après :

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <user defined native fault name>
  ```

  Si une application veut indiquer que le message de réponse entrant est une exception d'exécution, il doit y avoir un en-tête dans le message entrant comme indiqué ci-après :

  ```
  Header name = FaultType, Header value = Runtime
  ```

- Sélecteur d'erreurs SOAP

  Une erreur métier peut être envoyée dans le cadre d'un message SOAP avec l'en-tête SOAP personnalisé suivant : "CustomerAlreadyExists" est le nom de l'erreur dans ce cas.

  ```
  <ibmSoap:BusinessFaultName
  xmlns:ibmSoap="http://www.ibm.com/soap">CustomerAlreadyExists
  <ibmSoap:BusinessFaultName>
  ```

Le sélecteur d'erreurs est facultatif. Si vous ne spécifiez aucun sélecteur d'erreurs, la liaison d'importation ne peut pas déterminer le type de réponse. En conséquence, la liaison la traite comme une réponse métier et appelle le gestionnaire de données de réponse ou la liaison de données.

Vous pouvez créer un sélecteur d'erreurs personnalisé. Les étapes permettant de créer un sélecteur d'erreurs personnalisé se trouvent dans la rubrique «Developing a custom fault selector» du centre de documentation IBM Integration Designer.

Erreurs métier

Une erreur métier peut se produire lorsqu'il existe une erreur dans le traitement d'une requête. Par exemple, si vous envoyez une requête pour créer un client et que le client existe déjà, le service envoie une exception métier à la liaison d'importation.

Lorsqu'une exception métier est reçue par la liaison, les étapes du traitement dépendent de la configuration d'un sélecteur d'erreurs pour la liaison.

- Si aucun sélecteur d'erreurs n'a été configuré, la liaison appelle le gestionnaire de données de réponse ou la liaison de données.
- Si un sélecteur d'erreurs a été configuré, le traitement suivant se produit :
  1. La liaison d'importation appelle le sélecteur d'erreurs pour déterminer si la réponse est une erreur métier, une réponse métier ou une erreur d'exécution.
  2. Si la réponse est une erreur métier, la liaison d'importation appelle le sélecteur d'erreurs pour fournir le nom de l'erreur native.
  3. La liaison d'importation identifie l'erreur WSDL correspondant au nom de l'erreur native renvoyée par le sélecteur d'erreurs.

4. La liaison d'importation détermine le gestionnaire de données d'erreur configuré pour cette erreur WSDL.
5. La liaison d'importation appelle ce gestionnaire de données d'erreur avec les données d'erreur.
6. Le gestionnaire de données d'erreur transforme les données d'erreur en un objet de données et le renvoie à la liaison d'importation.
7. La liaison d'importation construit un objet d'exception métier de service avec l'objet de données et le nom de l'erreur.
8. L'importation renvoie l'objet d'exception métier de service au composant.

Exceptions d'exécution

Une exception d'exécution peut se produire lorsqu'il existe un problème de communication avec le service. Le traitement d'une exception d'exécution est identique au traitement d'une exception métier. Si un sélecteur d'erreurs a été configuré, le traitement suivant se produit :

1. La liaison d'importation appelle le gestionnaire de données d'exception d'exécution approprié avec les données d'exception.
2. Le gestionnaire de données d'exception d'exécution transforme les données d'exception en un objet d'exception d'exécution de service et le renvoie à la liaison d'importation.
3. L'importation renvoie l'objet d'exception d'exécution de service au composant.

Interopérabilité entre les modules SCA et les services Open SCA

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) offre un modèle de programmation simple, mais puissant, permettant de créer des applications basées sur les spécifications Open SCA. Les modules SCA de IBM Business Process Manager utilisent des liaisons d'importation et d'exportation pour interagir avec les services Open SCA développés dans un environnement Rational Application Developer et hébergés par WebSphere Application Server Feature Pack for Service Component Architecture.

Une application SCA appelle une application Open SCA par l'intermédiaire d'une liaison d'importation. Une application SCA reçoit un appel d'une application Open SCA par l'intermédiaire d'une liaison d'exportation. Une liste des liaisons prises en charge est présentée dans «Appel de services sur des liaisons interopérables», à la page 70.

Appel de services Open SCA à partir de modules SCA

Les applications SCA développées avec IBM Integration Designer peuvent appeler des applications Open SCA développées dans un environnement Rational Application Developer. Cette section illustre un exemple d'appel d'un service Open SCA à partir d'un module SCA à l'aide d'une liaison d'importation SCA.

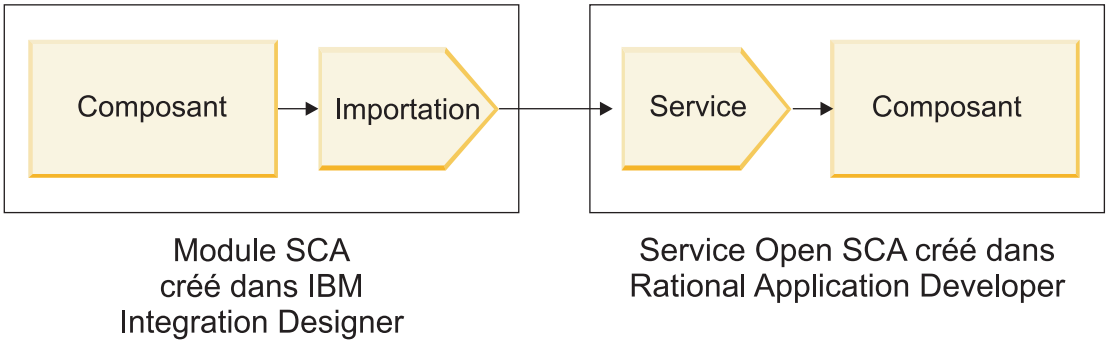


Figure 56. Composant de module SCA appelant un service Open SCA

Aucune configuration spéciale n'est requise pour appeler un service Open SCA.

Pour vous connecter à un service Open SCA par l'intermédiaire d'une liaison d'importation SCA, spécifiez le nom de composant et le nom de service du service Open SCA dans la liaison d'importation.

1. Pour obtenir le nom du composant cible et du service à partir du composite Open SCA, procédez comme suit :
   a. Vérifiez que la page **Propriétés** est ouverte en cliquant sur **Fenêtre** > **Afficher la vue** > **Propriétés**.
   b. Ouvrez l'éditeur de composite en cliquant deux fois sur le diagramme de composite qui contient le composant et le service. Par exemple, pour un composant intitulé **customer**, le diagramme du composite est **customer.composite_diagram**.
   c. Cliquez sur le composant cible.
   d. Dans la zone **Nom** de la page **Propriétés**, relevez le nom du composant cible.
   e. Cliquez sur l'icône de service associé au composant.
   f. Dans la zone **Nom** de la page **Propriétés**, relevez le nom du service.
2. Pour configurer l'importation IBM Business Process Manager afin de la connecter au service Open SCA, procédez comme suit :
   a. Dans IBM Integration Designer, accédez à l'onglet **Propriétés** de l'importation SCA à connecter au service Open SCA.
   b. Dans la zone **Nom du module**, entrez le nom de composant de l'étape 1d, à la page 69.
   c. Dans la zone **Nom d'exportation**, entrez le nom de service de l'étape 1f, à la page 69.
   d. Sauvegardez votre travail en appuyant sur les touches Ctrl+S.

Appel de modules SCA à partir de services Open SCA

Les applications Open SCA développées dans un environnement Rational Application Developer peuvent appeler des applications SCA développées avec IBM Integration Designer. Cette section illustre un exemple d'appel d'un module SCA (par l'intermédiaire d'une liaison d'exportation SCA) à partir d'un service Open SCA.

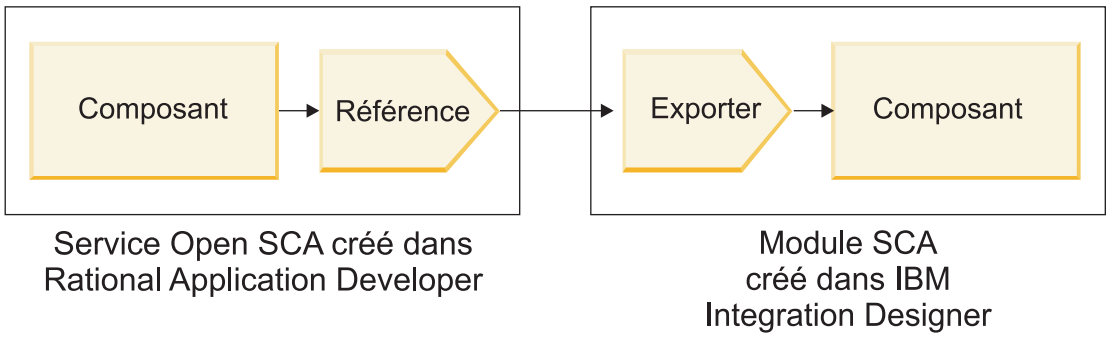


Figure 57. Service Open SCA appelant un composant dans un module SCA

Pour vous connecter à un composant SCA par l'intermédiaire d'une liaison de référence Open SCA, vous spécifiez le nom du module et le nom de l'exportation.

1. Pour obtenir le nom du composant cible et de l'exportation, procédez comme suit :
   a. Dans IBM Integration Designer, ouvrez le module dans l'éditeur d'assemblage en cliquant deux fois sur le module.
   b. Cliquez sur l'exportation.
   c. Dans la zone **Nom** de la page **Propriétés**, relevez le nom de l'exportation.
2. Configurez la référence Open SCA à connecter au module IBM Business Process Manager et à l'exportation :
   a. Dans Rational Application Developer, ouvrez l'éditeur de composite en cliquant deux fois sur le diagramme de composite qui contient le composant et le service.
   b. Cliquez sur l'icône de référence de la référence de composant pour en afficher les propriétés dans la page **Propriétés**.
   c. Cliquez sur l'onglet **Liaison** dans la partie gauche de la page.
   d. Cliquez sur **Liaisons**, puis sur **Ajouter**.
   e. Sélectionnez la liaison **SCA**.
   f. Dans la zone **Uri**, entrez le nom du module IBM Business Process Manager, suivi d'une barre oblique («/»), suivie du nom de l'exportation (que vous avez déterminé à l'étape 1c, à la page 70).
   g. Cliquez sur **OK**.
   h. Sauvegardez votre travail en appuyant sur les touches Ctrl+S.

Appel de services sur des liaisons interopérables

Les liaisons ci-après sont prises en charge pour l'interopérabilité avec un service Open SCA.

- Liaison SCA

  Dans IBM Business Process Manager, lorsqu'un module SCA appelle un service Open SCA par l'intermédiaire d'une liaison d'importation SCA, les styles d'appel suivants sont pris en charge :

  – Asynchrone (unidirectionnel)
  – Synchrone (demande/réponse)

  L'interface d'importation SCA et l'interface de service Open SCA doivent utiliser une interface WSDL compatible avec l'interopérabilité des services Web (WS-I).

  Notez que la liaison SCA prend en charge la propagation des transactions et des contextes de sécurité.
- Liaison de service Web (JAX-WS) avec protocole SOAP1.1/HTTP ou SOAP1.2/HTTP

  L'interface d'importation SCA et l'interface de service Open SCA doivent utiliser une interface WSDL compatible avec l'interopérabilité des services Web (WS-I).

  En outre, les qualités de service suivantes sont prises en charge :

  - Transaction atomique des services Web
  - Sécurité des services Web
- Liaison EJB

  Une interface Java est utilisée pour définir l'interaction entre un module SCA et un service Open SCA lorsque la liaison EJB est utilisée.

  Notez que la liaison EJB prend en charge la propagation des transactions et des contextes de sécurité.
- Liaisons JMS

  L'interface d'importation SCA et l'interface de service Open SCA doivent utiliser une interface WSDL compatible avec l'interopérabilité des services Web (WS-I).

  Les fournisseurs JMS pris en charge sont les suivants :
  - WebSphere Platform Messaging (Liaison JMS)
  - WebSphere MQ (Liaison JMS MQ)

Remarque : Les graphiques métier ne sont pas interopérables entre les liaisons SCA et ne sont donc pas prises en charge dans les interfaces utilisées pour interagir avec WebSphere Application Server Feature Pack for Service Component Architecture.

Types de liaison

Vous pouvez utiliser des *liaisons* spécifiques à un protocole avec des importations et des exportations pour spécifier les moyens de transport des données depuis ou vers un module.

Sélection des liaisons appropriées

Lorsque vous créez une application, vous devez savoir comment sélectionner la liaison la plus appropriée aux besoins de votre application.

Les liaisons disponibles dans IBM Integration Designer offrent une variété de choix. Cette liste permet d'identifier le type de liaison le plus approprié pour les besoins de votre application.

Choisissez une liaison *SCA (Service Component Architecture)* lorsque les facteurs suivants sont applicables :

- Tous les services sont contenus dans des modules, c'est-à-dire qu'il n'existe pas de services externes.
- Vous souhaitez séparer les fonctions dans des modules SCA différents qui interagissent directement entre eux.
- Les modules sont étroitement couplés.

Choisissez une liaison *service Web* lorsque ces facteurs sont applicables :

- Vous devez accéder à un service externe via Internet ou fournir un service via Internet.
- Les services sont faiblement couplés.
- Vous préférez les communications synchrones ; une demande d'un service peut attendre une réponse d'un autre service.
- Le protocole du service externe auquel vous accédez ou du service à fournir est SOAP/HTTP ou SOAP/JMS.

Choisissez une liaison *HTTP* si les facteurs suivants sont applicables :

- Vous devez accéder à un service externe via Internet ou fournir un service par ce même biais et vous utilisez d'autres services Web tels que GET, PUT et DELETE.
- Les services sont faiblement couplés.
- Vous préférez les communications synchrones ; une demande d'un service peut attendre une réponse d'un autre service.

Choisissez une liaison *EJB (Enterprise Java Beans)* si les facteurs suivants sont applicables :

- La liaison est destinée à un service importé qui est lui-même un EJB ou auquel les clients EJB doivent accéder.
- Le service importé est faiblement couplé.
- Les interactions EJB avec état ne sont pas requises.
- Vous préférez les communications synchrones ; une demande d'un service peut attendre une réponse d'un autre service.

Choisissez une liaison *EIS (Enterprise Information Systems)* si les facteurs suivants sont applicables :
- Vous devez accéder à un service sur un système EIS à l'aide d'un adaptateur de ressources.
- Vous préférez une transmission de données synchrone à une transmission asynchrone.

Choisissez une liaison *JMS (Java Message Service)* lorsque les facteurs suivants sont applicables :

Important : Il existe plusieurs types de liaison JMS. Si vous prévoyez d'échanger des messages SOAP à l'aide de JMS, choisissez la liaison de service Web avec le protocole SOAP/JMS. Voir «Liaisons de service Web», à la page 73.
- Vous devez accéder à un système de messagerie.
- Les services sont faiblement couplés.
- Vous préférez une transmission de données asynchrone à une transmission synchrone.

Choisissez une liaison *JMS (Generic Java Message Service)* lorsque les facteurs suivants sont applicables :
- Vous devez accéder à un système de messagerie d'un fournisseur non IBM.
- Les services sont faiblement couplés.
- La fiabilité est plus importante que les performances ; vous préférez une transmission de données asynchrone à une transmission synchrone.

Choisissez une liaison *MQ (Message Queue)* si les facteurs suivants sont applicables :
- Vous devez accéder à un système de messagerie WebSphere MQ et utiliser les fonctions natives de MQ.
- Les services sont faiblement couplés.
- La fiabilité est plus importante que les performances ; vous préférez une transmission de données asynchrone à une transmission synchrone.

Choisissez une liaison *JMS MQ* si les facteurs suivants sont applicables :
- Vous devez accéder à un système de messagerie WebSphere MQ, mais pouvez le faire dans un contexte JMS ; le sous-ensemble JMS de fonctions est suffisant pour votre application.
- Les services sont faiblement couplés.
- La fiabilité est plus importante que les performances ; vous préférez une transmission de données asynchrone à une transmission synchrone.

Liaisons SCA

Une liaison SCA (Service Component Architecture) permet à un service de communiquer avec d'autres services dans d'autres modules. Une importation avec une liaison SCA permet d'accéder à un service dans un autre module SCA. Une exportation avec une liaison SCA permet d'offrir un service à d'autres modules.

Utilisez IBM Integration Designer pour générer et configurer des liaisons de service Web pour les importations et exportations dans des modules SCA.

Si des modules sont exécutés sur le même serveur ou déployés dans le même cluster, une liaison SCA représente la liaison la plus simple et la plus rapide à utiliser.

Une fois que le module contenant la liaison SCA est déployé sur le serveur, vous pouvez utiliser la console d'administration pour afficher des informations sur la liaison ou, dans le cas d'une liaison d'importation, pour changer les propriétés sélectionnées de cette liaison.

Liaisons de service Web

Une liaison de service Web représente le moyen par lequel les messages sont transmis d'un composant SCA (Service Component Architecture) vers un service Web (et inversement).

Présentation des liaisons de service Web : Advanced

Une liaison d'importation de service Web vous permet d'appeler un service Web externe depuis vos composants SCA (Service Component Architecture). Une liaison d'exportation de service Web vous permet d'exposer vos composants SCA aux clients en tant que services Web.

Avec une liaison de service Web, vous pouvez accéder à des services externes via des messages SOAP interopérables et des qualités de service (QoS).

Utilisez Integration Designer pour générer et configurer des liaisons de service Web pour les importations et exportations dans des modules SCA. Les types suivants de liaisons de service Web sont disponibles :

- SOAP1.2/HTTP et SOAP1.1/HTTP

  Ces liaisons se basent sur JAX-WS (Java API for XML Web Services), une API de programmation Java permettant de créer des services Web.

  – Utilisez SOAP1.2/HTTP si votre service Web respecte la spécification SOAP 1.2.
  – Utilisez SOAP1.1/HTTP si votre service Web respecte la spécification SOAP 1.1.

  Important : Lorsque vous déployez une application avec une liaison de service Web (JAX-WS), vérifiez que l'option **Démarrer les composants en fonction des besoins** n'est pas sélectionnée dans le serveur cible. Pour plus de détails, voir «Vérification de la configuration du serveur», à la page 82.

  Lorsque vous sélectionnez l'une de ces liaisons, vous pouvez envoyer des pièces jointes avec vos messages SOAP.

  Les liaisons de service Web sont compatibles avec les messages SOAP standard. Toutefois, en utilisant l'une des liaisons JAX-WS de service Web, vous pouvez personnaliser la manière dont les messages SOAP sont analysés ou écrits. Par exemple, vous pouvez gérer les éléments non standard dans des messages SOAP ou appliquer un traitement supplémentaire au message SOAP. Lorsque vous configurez la liaison, vous spécifiez un gestionnaire de données personnalisé qui effectue ce traitement sur le message SOAP.

  Vous pouvez utiliser des ensembles de règles avec une liaison de service Web (JAX-WS). Un ensemble de règles regroupe différents types de règles qui procurent chacun une qualité de service définie (QoS). Par exemple, l'ensemble de règles WSAddressing permet d'adresser des services web et des messages de manière uniforme et indépendamment des systèmes de transport. Utilisez Integration Designer pour sélectionner l'ensemble de règles pour la liaison.

  Remarque : Pour utiliser un ensemble de règles SAML (Security Assertion Markup Language), vous devez effectuer certaines configurations supplémentaires, comme décrit dans «Importation des ensembles de règles SAML», à la page 79.

- SOAP1.1/HTTP

  Utilisez cette liaison si vous souhaitez créer des services Web qui utilisent un message codé par SOAP en fonction de la spécification JAX-RPC (Java API for XML-based RPC).

- SOAP1.1/JMS

  Utilisez cette liaison pour envoyer ou recevoir des messages SOAP à l'aide d'une destination JMS (Java Message Service).

Quel que soit le transport (HTTP ou JMS) utilisé pour transmettre les message SOAP, les liaisons de service Web traitent toujours les interactions de demande/réponse de manière synchrone. L'unité d'exécution à l'origine de l'appel sur le fournisseur de services est bloquée jusqu'à la réception d'une réponse du fournisseur. Pour plus d'informations sur ce style d'appel, voir «Appel synchrone».

Important : Les combinaisons de liaisons de services web suivantes ne peuvent pas être utilisées sur les exportations sur le même module. Si vous devez exposer des composants à l'aide de plusieurs de ces liaisons d'exportation, vous devez placer chacun dans un module distinct, puis connecter ces modules à vos composants à l'aide de la liaison SCA :

- SOAP 1.1/JMS et SOAP 1.1/HTTP à l'aide de JAX-RPC
- SOAP 1.1/HTTP à l'aide de JAX-RPC et SOAP 1.1/HTTP à l'aide de JAX-WS
- SOAP 1.1/HTTP à l'aide de JAX-RPC et SOAP 1.2/HTTP à l'aide de JAX-WS

Une fois que le module SCA contenant la liaison de service Web est déployé sur le serveur, vous pouvez utiliser la console d'administration pour afficher des informations sur la liaison ou pour modifier les propriétés sélectionnées de cette liaison.

Remarque : Les services Web permettent aux applications d'interopérer en utilisant les descriptions standard des services et les formats standard des messages qu'ils échangent. Par exemple, les liaisons d'importation et d'exportation de services Web peuvent interopérer avec des services implémentés à l'aide de Web Services Enhancements (WSE) Version 3.5 et Windows Communication Foundation (WCF) Version 3.5 pour Microsoft .NET. Lors de l'interopération avec de tels services, vous devez veiller à ce que les conditions suivantes soient remplies :

- Le fichier WSDL (Web Services Description Language) utilisé pour accéder à une exportation de service Web inclut une valeur d'action SOAP non vide pour chaque opération exposée dans l'interface.
- Le client de service Web définit l'en-tête SOAPAction ou l'en-tête wsa:Action lorsqu'il envoie des messages à une exportation de service web.

Propagation d'en-têtes SOAP : 

Lors de la gestion des messages SOAP, vous devez accéder aux informations de certains en-têtes SOAP dans les messages reçus. Vérifiez que des messages avec des en-têtes SOAP sont envoyés avec des valeurs spécifiques ou autorisez ces en-têtes SOAP à passer par un module.

Lorsque vous configurez une liaison de services Web dans Integration Designer, vous pouvez indiquer que les en-têtes de SOAP doivent être propagés.

- Lorsque des demandes sont reçues lors d'une exportation ou des réponses sont reçues lors d'une importation, les informations sur les en-têtes SOAP sont accessibles, ce qui permet à la logique dans un module de se baser sur les valeurs d'en-têtes et à ces en-têtes d'être modifiés.
- Lorsque des demandes sont envoyées d'une exportation ou des réponses sont envoyées d'une importation, des en-têtes SOAP peuvent être inclus dans ces messages.

La forme et la présence des en-têtes SOAP propagés peuvent être affectées par les ensembles de règles configurés dans l'importation ou l'exportation, comme décrit dans le tableau 31, à la page 76.

Pour configurer la propagation des en-têtes SOAP pour une importation ou une exportation, vous cliquez (dans la vue Propriétés de Integration Designer) dans l'onglet **Propager l'en-tête de protocole** et sélectionnez les options dont vous avez besoin.

En-tête WS-Addressing

L'en-tête WS-Addressing peut être propagé par la liaison de service Web (JAX-WS).

Quand vous propagez l'en-tête WS-Addressing, prenez en compte les informations suivantes :

- Si vous activez la propagation pour l'en-tête WS-Addressing, celui-ci est propagé dans le module dans les cas suivants :
  - si des demandes sont reçues dans une exportation,
  - si des réponses sont reçues dans une importation.
- L'en-tête WS-Addressing n'est pas propagé dans des messages sortants de IBM Business Process Manager (l'en-tête n'est pas propagé quand des demandes sont envoyées d'une importation ou quand des réponses sont envoyées d'une exportation).

En-tête WS-Security

L'en-tête WS-Security peut être propagé par la liaison de service Web (JAX-WS) et la liaison de service Web (JAX-RPC).

La spécification WS-Security des services Web décrit les améliorations de la messagerie SOAP afin d'offrir une qualité de protection via l'intégrité des messages, leur confidentialité et l'authentification des messages. Ces mécanismes peuvent permettre d'accepter toute une gamme de modèles de sécurité et de technologies de chiffrement.

Quand vous propagez l'en-tête WS-Security, prenez en compte les informations suivantes :

- Si vous activez la propagation pour l'en-tête WS-Security, celui-ci est propagé à travers le module dans les cas suivants :
  - si des demandes sont reçues dans une exportation,
  - si des demandes sont envoyées d'une importation,
  - si des réponses sont reçues dans une importation.
- Par défaut, l'en-tête n'est *pas* propagé quand des réponses sont envoyées d'une exportation Toutefois, si vous définissez la propriété JVM **WSSECURITY.ECHO.ENABLED** à **true**, l'en-tête est propagé quand des réponses sont envoyées de l'exportation. Dans ce cas, si l'en-tête WS-Security dans le chemin de demande n'est pas modifié, les en-têtes WS-Security peuvent être automatiquement répercutés de demandes en réponses.
- La forme exacte du message SOAP envoyé d'une importation pour une demande ou d'une exportation pour une réponse peut ne pas exactement correspondre au message SOAP reçu à l'origine. C'est pourquoi toute signature numérique est censée devenir invalide. Si une signature numérique est obligatoire dans les messages envoyés, elle doit être établie à l'aide d'un ensemble de règles de sécurité approprié, et les en-têtes relatifs à la signature numérique dans les messages reçus doivent être supprimés dans le module.

Pour propager l'en-tête WS-Security, vous devez inclure le schéma WS-Security avec le module d'application. Voir «Inclusion du schéma WS-Security dans un module d'application», à la page 76 pour connaître la procédure pour inclure le schéma.

Mode de propagation des en-têtes

Le mode de propagation des en-têtes dépend de la définition des règles de sécurité sur la liaison d'importation ou d'exportation, comme décrit dans le tableau 31, à la page 76 :

Tableau 60. Mode de transmission des en-têtes de sécurité

	Liaison d'exportation sans règle de sécurité	**Liaison d'exportation avec règle de sécurité**
Liaison d'importation sans règle de sécurité	Les en-têtes de sécurité sont transmis en l'état via le module. Ils ne sont pas déchiffrés. Les en-têtes sont envoyés en sortie sous la même forme qu'ils ont été reçus. La signature numérique peut devenir invalide.	Les en-têtes de sécurité sont déchiffrés et transmis via le module avec la vérification et l'authentification de la signature. Les en-têtes déchiffrés sont envoyés en sortie. La signature numérique peut devenir invalide.
Liaison d'importation avec règle de sécurité	Les en-têtes de sécurité sont transmis en l'état via le module. Ils ne sont pas déchiffrés. Les en-têtes ne doivent pas être propagés vers l'importation. Sinon, une erreur se produit en raison de la duplication.	Les en-têtes de sécurité sont déchiffrés et transmis via le module avec la vérification et l'authentification de la signature. Les en-têtes ne doivent pas être propagés vers l'importation. Sinon, une erreur se produit en raison de la duplication.

Configurez les ensembles de règles appropriés sur les liaisons d'exportation et d'importation, car l'opération épargne le demandeur de service des modifications apportées à la configuration ou aux exigences QoS du fournisseur de services. Si des en-têtes SOAP standard sont visibles dans un module, ils peuvent être utilisés pour influencer le traitement (par exemple, la consignation et le traçage) dans le module. La propagation des en-têtes SOAP à travers le module, d'un message reçu à un message envoyé, signifie que les avantages de l'isolement du module sont moindres.

Les en-têtes standard, comme les en-têtes WS-Security, ne doivent pas être propagés sur une demande vers une importation ou sur une réponse vers une exportation lorsqu'un ensemble de règles, qui entraînerait normalement la génération de ces en-têtes, est associé à une importation ou une exportation. Sinon, une erreur se produit en raison d'une duplication de ces en-têtes. A la place, les en-têtes doivent être explicitement supprimés, ou bien la liaison d'importation ou d'exportation doit être configurée pour empêcher la propagation des en-têtes de protocole.

Accès aux en-têtes SOAP

Quand un message contenant des en-têtes est reçu d'une importation ou d'une exportation de service Web, les en-têtes sont placés dans la section d'en-têtes de l'objet de message de service (SMO). Vous pouvez accéder aux informations d'en-tête, comme décrit dans «Accès aux informations d'en-tête SOAP dans le SMO».

Inclusion du schéma WS-Security dans un module d'application

La procédure suivante montre les étapes pour inclure le schéma dans le module d'application :

- Si l'ordinateur sur lequel Integration Designer s'exécute a accès à Internet, procédez comme suit :
  1. Dans la perspective Intégration métier, sélectionnez **Dépendances** pour votre projet.
  2. Développez **Ressources prédéfinies** et sélectionnez **Fichiers de schéma WS-Security 1.0** ou **Fichiers de schéma WS-Security 1.1** pour importer le schéma dans votre module.
  3. Effacez et régénérez le projet.
- Si l'ordinateur sur lequel Integration Designer s'exécute n'a pas accès à Internet, vous pouvez télécharger le schéma sur un autre ordinateur ayant accès à Internet. Vous pouvez ensuite le copier sur l'ordinateur où Integration Designer s'exécute.

1. Depuis l'ordinateur qui a accès à Internet, téléchargez le schéma distant :
   a. Cliquez sur **Fichier** > **Importer** > **Intégration métier** > **WSDL et XSD**.
   b. Sélectionnez **WSDL distant** ou **Fichier XSD**.
   c. Importez les schémas suivants :
      http://www.w3.org/2003/05/soap-envelope/
      http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd
      http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
2. Copiez les schémas sur l'ordinateur sans accès à Internet.
3. Depuis l'ordinateur sans accès à Internet, importez le schéma :
   a. Cliquez sur **Fichier** > **Importer** > **Intégration métier** > **WSDL et XSD**.
   b. Sélectionnez **WSDL local** ou **Fichier XSD**.
4. Modifiez les emplacements des schémas pour oasis-wss-wssecurity_secext-1.1.xsd :
   a. Ouvrez le schéma dans *emplacement_espace_de_travail/nom_module*/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd.
   b. Remplacez :
      ```
      <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
      schemaLocation='http://www.w3.org/2003/05/soap-envelope/'/>
      ```
      par :
      ```
      <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
      schemaLocation='../w3/_2003/_05/soap_envelope.xsd'/>
      ```
   c. Remplacez :
      ```
      <xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
      schemaLocation='http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd'/>
      ```
      par :
      ```
      <xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
      schemaLocation='../w3/tr/_2002/rec_xmlenc_core_20021210/xenc-schema.xsd'/>
      ```
5. Modifiez l'emplacement du schéma pour oasis-200401-wss-wssecurity-secext-1.0.xsd :
   a. Ouvrez le schéma dans *emplacement_espace_de_travail/nom_module*/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd.
   b. Remplacez :
      ```
      <xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
      schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
      ```
      par :
      ```
      <xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
      schemaLocation="../w3/tr/_2002/rec_xmldsig_core_20020212/xmldsig-core-schema.xsd"/>
      ```
6. Effacez et régénérez le projet.

Propagation d'en-têtes de transport : Advanced

Lors de la gestion des messages SOAP, vous devez accéder aux informations de certains en-têtes de transport dans les messages reçus. Vérifiez que des messages avec des en-têtes de transport sont envoyés avec des valeurs spécifiques ou autorisez ces en-têtes de transport à passer par un module.

Lorsque vous configurez une liaison de services Web dans Integration Designer, vous pouvez indiquer que les en-têtes de transport doivent être propagés.

- Lorsque des demandes sont reçues lors d'une exportation ou des réponses sont reçues lors d'une importation, les informations sur les en-têtes de transport sont accessibles, ce qui permet à la logique dans un module de se baser sur les valeurs d'en-têtes et à ces en-têtes d'être modifiés.
- Lorsque des réponses sont envoyées d'une exportation ou des demandes sont envoyées d'une importation, des en-têtes de transport peuvent être inclus dans ces messages.

Spécification d'une propagation d'en-têtes

Pour configurer la propagation des en-têtes de transport pour une importation ou une exportation, procédez comme suit :

1. Dans la vue Propriétés de Integration Designer, sélectionnez **Liaison** > **Propagation**.
2. Définissez l'option de propagation d'en-têtes de votre choix.

Remarque : Par défaut, la propagation d'en-têtes de transport est désactivée et ne peut être déployée que dans un environnement d'exécution Version 7.0.0.3 (ou ultérieure). Notez aussi que pour la Version 7.0.0.3, la propagation d'en-têtes de transport est limitée à des en-têtes de transport HTTP uniquement.

Si vous activez une propagation des en-têtes de transport, les en-têtes seront propagés sur un module à partir de messages reçus et, si vous ne supprimez pas explicitement les en-têtes, ceux-ci seront utilisés dans les appels suivants dans la même unité d'exécution.

Remarque : Les en-têtes de transport ne peuvent pas être propagés si vous utilisez la liaison de services web (JAX-RPC).

Accès aux informations d'en-tête

Si la propagation d'en-têtes de transport est activée pour des messages reçus, tous les en-têtes de transport (y compris des en-têtes définis par le client) sont visibles dans l'objet de message de service (SMO). Vous pouvez définir les en-têtes pour différentes valeurs ou en créer d'autres. Notez toutefois qu'il n'existe aucun contrôle ni validation des valeurs que vous définissez et que tout en-tête incorrect peut causer des problèmes lors de l'exécution du service Web.

Prenez en compte les informations suivantes lorsque vous définissez des en-têtes HTTP :

- Tout changement apporté aux en-têtes réservés pour un moteur de service Web ne sera pas repris dans le message sortant. Par exemple, la version ou méthode HTTP, le contenu type, la longueur du contenu et les en-têtes SOAPAction sont réservés au moteur de service Web.
- Si la valeur d'en-tête est un nombre, le nombre (contrairement à la chaîne) doit être défini directement. Par exemple, utilisez **Max-Forwards = 5** (au lieu de **Max-Forwards = Max-Forwards: 5**) et **Age = 300** (au lieu de **Age = Age: 300**).
- Si la taille du message de demande ne dépasse pas 32 Ko, le moteur de services Web supprime l'en-tête Transfert-Codage et définit à la place l'en-tête Longueur-contenu avec la taille fixe du message.
- Le Langage-Contenu est réinitialisé par WAS.channel.http sur le chemin de réponse.
- Un paramètre de mise à niveau non valide se traduit par une erreur 500.
- Les en-têtes suivants ajoutent la valeur réservée par le moteur de service Web aux paramètres du client :
  - Agent d'utilisateur
  - Contrôle de cache
  - Pragma
  - Accepter
  - Connexion

Vous pouvez accéder aux informations d'en-tête de l'une des manières suivantes :

- En utilisant une primitive de médiation pour accéder à la structure SMO.

  Voir les liens «Rubriques connexes» pour retrouver des informations sur l'utilisation des primitives de médiation.
- En utilisant une interface SPI des services de contexte

  Le code exemple suivant lit des en-têtes de transport HTTP à partir de services de contexte :

```
HeadersType headerType = ContextService.INSTANCE.getHeaders();
HTTPHeaderType httpHeaderType = headerType.getHTTPHeader();
List HTTPHeader  httpHeaders = httpHeaderType.getHeader();
if(httpHeaders!=null){
 for(HTTPHeader httpHeader: httpHeaders){
   String httpHeadername = httpHeader.getName();
   String httpHeaderValue = httpHeader.getValue();
     }
}
List PropertyType  properties = headerType.getProperties();
if(properties!=null){
  for(PropertyType property: properties){
   String propertyName = property.getName();
   String propertyValue = property.getValue().toString();
     }
}
```

Identification et résolution des incidents

En cas de problèmes lors de l'envoi des en-têtes révisés, vous pouvez intercepter le message TCP/IP en utilisant des outils comme Moniteur TCP/IP dans Integration Designer. Pour accéder au Moniteur TCP/IP, dans la page Préférence, sélectionnez **Run/Debug** > **Moniteur TCP/IP**.

Vous pouvez aussi visualiser les valeurs d'en-tête à l'aide de la trace du moteur JAX-WS : **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Utilisation des liaisons de services Web (JAX-WS) : Advanced

Lorsque vous utilisez des liaisons de services Web (JAX-WS) avec vos applications, vous pouvez ajouter une qualité de service (QOS) SAML (Security Assertion Markup Language) à la liaison. Vous devez tout d'abord utiliser la console d'administration pour importer l'ensemble de règles. Vous pouvez également utiliser la console d'administration pour vous assurer que le serveur est correctement configuré pour être utilisé avec la liaison de services Web (JAX-WS).

Importation des ensembles de règles SAML : Advanced

Norme OASIS basée sur XML, SAML (Security Assertion Markup Language) permet l'échange d'informations entre des attributs d'identité de l'utilisateur et de sécurité. Lorsque vous configurez une liaison de services Web (JAX-WS) dans Integration Designer, vous pouvez indiquer un ensemble de règles SAML. A l'aide de la console d'administration d'IBM Business Process Manager, commencez par rendre les ensembles de règles disponibles pour permettre leur importation dans Integration Designer.

Les ensembles de règles SAML se trouvent généralement dans le répertoire de configuration des profils :

racine_profil/config/templates/PolicySets

Avant de lancer cette procédure, vérifiez que les répertoires suivants (qui contiennent les ensembles de règles) se trouvent bien dans le répertoire de configuration des profils :

- SAML11 Bearer WSHTTPS default
- SAML20 Bearer WSHTTPS default
- SAML11 Bearer WSSecurity default
- SAML20 Bearer WSSecurity default
- SAML11 HoK Public WSSecurity default
- SAML20 HoK Public WSSecurity default
- SAML11 HoK Symmetric WSSecurity default
- SAML20 HoK Symmetric WSSecurity default

- `Username WSHTTPS default`

Si les répertoires ne figurent pas dans le répertoire de configuration des profils, copiez-les dans ce répertoire à partir de l'emplacement suivant :

racine_serveur_app`/profileTemplates/default/documents/config/templates/PolicySets`

Importez les ensembles de règles dans la console d'administration, sélectionnez ceux que vous voulez rendre accessibles à Integration Designer, pour sauvegardez un fichier .zip pour chacun de ces ensembles de règles dans un emplacement accessible par Integration Designer.

1. Importez les ensembles de règles en procédant comme suit :
   a. Dans la console d'administration, cliquez sur **Services** > **Ensemble de règles** > **Ensembles de règles de l'application**.
   b. Cliquez sur **Importer** > **A partir du référentiel par défaut**.
   c. Sélectionnez l'ensemble de règles par défaut SAML, puis cliquez sur **OK**.
2. Exportez les ensembles de règles pour qu'ils puissent être utilisés par Integration Designer:
   a. Sur la page Ensembles de règles de l'application, sélectionnez l'ensemble de règles SAML à exporter, puis cliquez sur **Exporter**.

      Remarque : Si la page Ensembles de règles de l'application n'est pas actuellement affichées, sur la console d'administration, cliquez sur **Services** > **Ensembles de règles** > **Ensembles de règles de l'application**.
   b. Sur la page suivante, cliquez sur le lien du fichier .zip pour l'ensemble de règles.
   c. Dans la fenêtre de téléchargement de fichier, cliquez sur **Sauvegarder** et indiquez un emplacement accessible par Integration Designer.
   d. Cliquez sur **Retour**.
   e. Complétez les étapes 2a, à la page 80 via 2d, à la page 80 pour chaque ensemble de règles à exporter.

Les ensembles de règles sont enregistrés dans des fichiers .zip et sont prêts à être exportés dans Integration Designer.

Importez les ensembles de règles dans Integration Designer, comme décrit dans la rubrique «Ensembles de règles».

Appel des services Web utilisant l'authentification de base HTTP : 

Le procédure d'authentification de base HTTP demande de saisir un nom d'utilisateur et un mot de passe pour authentifier un client de service sur un noeud final sécurisé. Vous pouvez configurer l'authentification de base HTTP lorsque vous envoyez ou que vous recevez des requêtes de service Web.

Pour configurer l'authentification de base HTTP afin de recevoir des requêtes de service Web, vous devez configurer la liaison d'exportation de l'API Java des services Web XML (JAX-WS) comme indiqué dans Création et affectation de rôles de sécurité aux exportations de service Web.

Vous pouvez activer l'authentification de base HTTP pour les requêtes de service Web envoyées par une liaison d'importation de l'API Java des services Web XML de deux manières :

- Lorsque vous configurez la liaison d'importation dans un module SCA, vous pouvez sélectionner l'ensemble de règles HTTP nommé BPMHTTPBasicAuthentication (fourni avec la liaison d'importation de service Web JAX-WS) ou un autre ensemble de règles contenant la règle HTTPTransport.
- Lorsque vous construisez le module SCA, vous pouvez utiliser des fonctions de flux de médiation pour créer dynamiquement un en-tête d'authentification HTTP et spécifier les données du nom d'utilisateur et du mot de passe dans cet en-tête.

Remarque : L'ensemble de règles est prioritaire sur la valeur indiquée dans l'en-tête. Si vous souhaitez utiliser la valeur définie dans l'en-tête d'authentification HTTP pendant la phase d'exécution, n'associez pas un ensemble de règles comprenant la règle HTTPTransport. En particulier, n'utilisez pas l'ensemble de règles par défaut BPMHTTPBasicAuthentication et, si vous avez défini un autre ensemble de règles, vérifiez qu'il ne comprend pas la règle HTTPTransport.

Pour plus d'informations sur les liaisons de règles, les ensembles de règles de service web et leur utilisation, voir Ensembles de règles de service Web du centre de documentation de WebSphere Application Server.

- Pour utiliser l'ensemble de règles fourni, exécutez les opérations suivantes :
  1. Facultatif : Dans la console d'administration, créez une liaison de règles générales de client ou éditez une liaison de règles existante qui contienne la règle HTTPTransport avec les valeurs requises pour l'ID utilisateur et le mot de passe.
  2. Dans IBM Integration Designer, générez une liaison d'importation de service Web (JAX-WS) et associez-y l'ensemble de règles BPMHTTPBasicAuthentication.
  3. Exécutez l'*une* des actions suivantes :
     - Dans IBM Integration Designer, dans les propriétés de la liaison d'importation de service Web (JAX-WS), indiquez le nom d'une liaison de règles générales de client existante comprenant la règle HTTPTransport.
     - Après avoir déployé le module SCA, utilisez la console d'administration pour sélectionner une liaison de règles de client existante ou créer une nouvelle liaison de règles de client et l'associer à la liaison d'importation.
  4. Facultatif : Dans la console d'administration du Process Server, éditez la liaison d'ensemble de règles pour y spécifier l'ID et le mot de passe requis.
- Pour indiquer le nom d'utilisateur et le mot de passe dans l'en-tête d'authentification HTTP, exécutez l'une des procédures suivantes :
  - Utilisez la primitive de médiation HTTP Header Setter dans IBM Integration Designer afin de créer l'en-tête d'authentification HTTP, puis indiquez le nom d'utilisateur et le mot de passe.
  - S'il faut ajouter une logique supplémentaire, insérez un code Java dans une primitive de médiation personnalisée (comme dans l'exemple suivant) pour :
    1. Créer l'en-tête d'authentification HTTP
    2. Indiquer les données du nom d'utilisateur et du mot de passe
    3. Ajouter le nouvel en-tête d'authentification HTTP à la règle HTTPControl
    4. Redéfinir la règle HTTPControl après sa mise à jour dans le service de contexte

```
//Get the HeaderInfoType from contextService
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Get the HTTP header and HTTP Control from HeaderInfoType
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Create new HTTPAuthentication and set the HTTPCredentials
HTTPAuthentication authorization = factory.createHTTPAuthentication();
HTTPCredentials credentials = factory.createHTTPCredentials();
authorization.setAuthenticationType(HTTPAuthenticationType.BASIC_LITERAL);
credentials.setUserId("USERNAME");
credentials.setPassword("PASSWORD");
authorization.setCredentials(credentials);
```

```
cp.setAuthentication(authorization);
httpHeaderType.setControl(cp);
// Set header info back to the current execution context.
contextService.setHeaderInfo(headers);
```

Vérification de la configuration du serveur : Advanced

Lorsque vous déployez une application avec une liaison de service Web (JAX-WS), vérifiez que l'option **Démarrer les composants en fonction des besoins** du serveur cible sur laquelle l'application est déployée n'est pas sélectionnée.

Pour vérifier cela, procédez comme suit à partir de la console d'administration :

1. Cliquez sur **Serveurs** > **Types de serveurs** > **Serveurs d'applications WebSphere**.
2. Cliquez sur le nom du serveur.
3. Sous l'onglet Configuration, vérifiez si l'option **Démarrer les composants en fonction des besoins** est sélectionnée.
4. Exécutez l'une des étapes suivantes :
   - Si l'option **Démarrer les composants en fonction des besoins** est sélectionnée, désactivez la case à cocher, puis cliquez sur **Appliquer**.
   - Si l'option **Démarrer les composants en fonction des besoins** n'est pas sélectionnée, cliquez sur **Annuler**.

Pièces jointes dans les messages SOAP : Advanced

Vous pouvez envoyer et recevoir des messages SOAP qui incluent des données binaires (fichiers PDF ou images JPEG) comme pièces jointes. Les pièces jointes peuvent être *référencées* (à savoir, représentées explicitement comme parties de message dans l'interface de service) ou *non référencées* (auquel cas, un nombre et des types arbitraires de pièces jointes peuvent être incluses).

Une pièce jointe référencée peut être représentée de l'une des manières suivantes :

- Les pièces jointes MTOM utilisent le codage SOAP spécifié par Message Transmission Optimization Mechanism (http://www.w3.org/TR/soap12-mtom/). Les pièces jointes MTOM sont activées par l'intermédiaire d'une option de configuration dans les liaisons d'importation et d'exportation, et doivent être utilisées pour coder les pièces jointes des nouvelles applications.
- Comme élément wsi:swaRef-typed dans le schéma du message

  Les pièces jointes définies à l'aide du type wsi:swaRef sont conformes à Web Services Interoperability Organization (WS-I) *Attachments Profile Version 1.0* (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), qui définit la manière dont les éléments de message sont associés aux composants MIME.
- Comme composant de message de niveau supérieur, à l'aide d'un schéma de type binaire

  Les pièces jointes représentées sous forme de composants de message de niveau supérieur respectent la spécification des *messages SOAP avec pièces jointes* (http://www.w3.org/TR/SOAP-attachments).

  Vous pouvez aussi configurer des pièces jointes représentées comme des parties de message de niveau supérieur pour vous assurer que le document WSDL et les messages produits par la liaison sont conformes à WS-I *Attachments Profile Version 1.0* et à WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

Une pièce jointe non référencée est transportée dans un message SOAP sans aucune représentation dans le schéma du message.

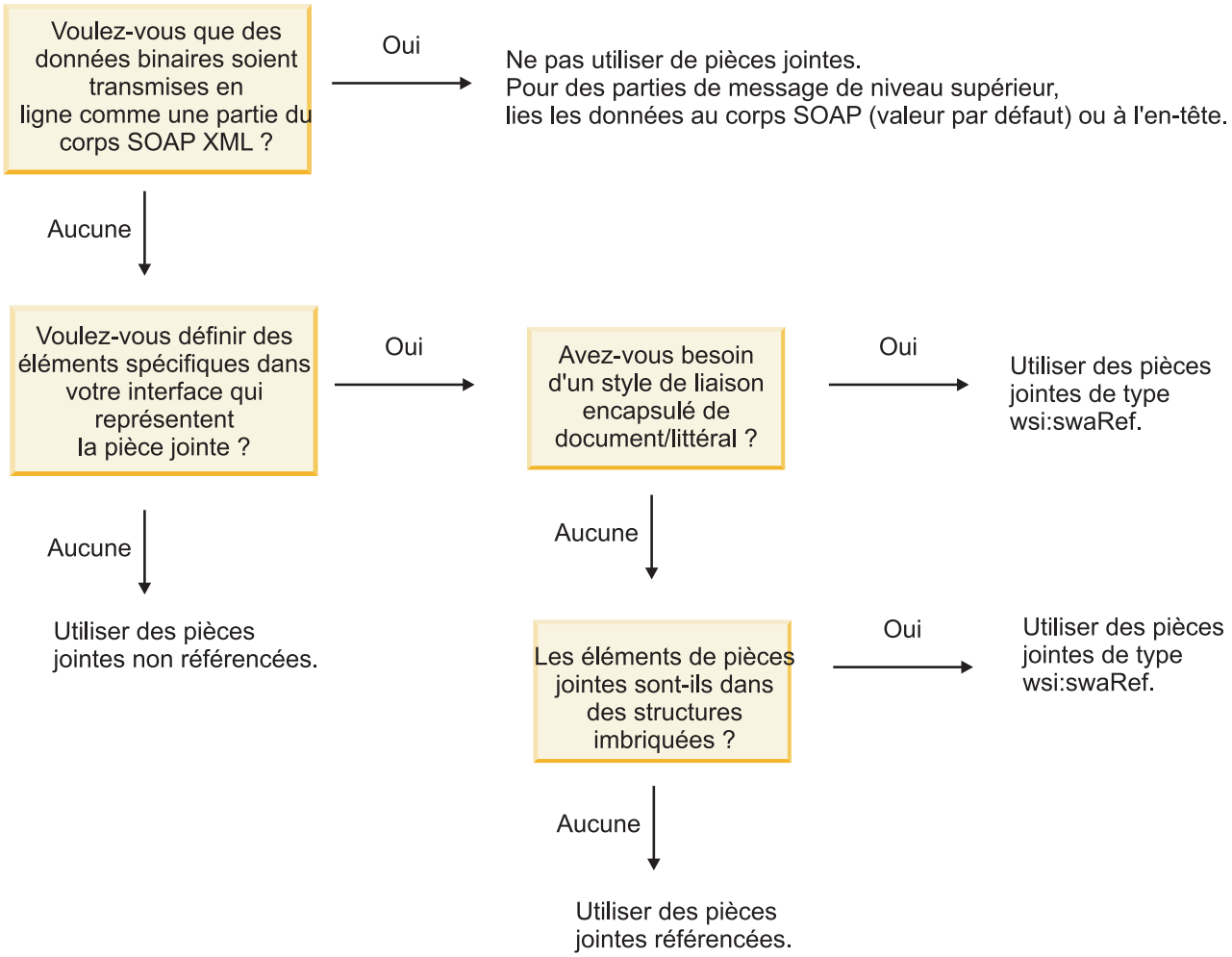
Dans tous les cas, sauf pour les pièces jointes MTOM, la liaison SOAP WSDL doit inclure une liaison MIME pour que des pièces jointes puissent être utilisées et la taille maximale de ces pièces jointes ne doit pas dépasser 20 Mo.

Remarque : Pour envoyer ou recevoir des messages SOAP avec des pièces jointes, vous devez utiliser l'une des liaisons de service Web basée sur JAX-WS (Java API for XML Web Services).

Choix du style de pièces jointe appropriées : Advanced

Lors de la conception d'une interface de service qui inclut des données binaires, envisagez la façon dont les données binaires sont traitées dans les messages SOAP qui sont envoyés et reçus par le service.

Le mécanisme MTOM (Message Transmission Optimization Mechanism) doit être utilisé pour les pièces jointes si l'application de service Web connectée le prend en charge. Si tel n'est pas le cas, le diagramme suivant indique comment les autres styles de pièces jointes sont choisis. A l'aide du jeu de questions suivant déterminez le style de pièces jointes approprié :



Pièces jointes MTOM : composants de message de niveau supérieur :

Vous pouvez envoyer et recevoir des messages de service Web qui incluent des pièces jointes SOAP MTOM (Message Transmission Optimization Mechanism). Dans un message SOAP à plusieurs parties MIME, le corps SOAP constitue la première partie du message et la ou les pièces jointes sont dans les parties suivantes.

Lorsque vous envoyez ou recevez une pièce jointe référencée dans un message SOAP, les données binaires qui constituent la pièce jointe (dont la taille est souvent importante) sont conservées séparément

du corps du message SOAP pour qu'elles n'aient pas besoin d'être analysées comme des données XML. Le traitement est donc plus efficace que si les données binaires étaient conservées dans un élément XML.

Voici un exemple de message SOAP MTOM :

```
... autres en-têtes de transport ...
Content-Type: multipart/related; boundary=MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812;
type="application/xop+xml"; start="
<0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>"; start-info="text/xml"; charset=UTF-8

--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: application/xop+xml; charset=UTF-8; type="text/xml";
content-transfer-encoding: binary
content-id:
   <0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header/>
      <soapenv:Body>
         <sendImage xmlns="http://org/apache/axis2/jaxws/sample/mtom">
            <input>
            <imageData><xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
            href="cid:1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org"/></imageData>
            </input>
         </sendImage>
      </soapenv:Body>
   </soapenv:Envelope>
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: text/plain
content-transfer-encoding: binary
content-id:
         <1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org>

... emplacement pour les données binaires ...
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812--
```

Notez que, dans cet exemple de message MTOM, l'élément content-type de l'enveloppe SOAP est **application/xop+xml** et que les données binaires sont remplacées par un élément **xop:Include** similaire au suivant :

```
<xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
href="cid:1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org"/>
```

Traitement entrant des pièces jointes référencées

Lorsqu'un client transmet un message SOAP avec une pièce jointe à un composant SCA (Service Component Architecture), la liaison d'exportation de service Web (JAX-WS) commence par supprimer cette pièce jointe. Elle analyse ensuite la partie SOAP du message et crée un objet métier. Enfin, la liaison définit le code binaire de la pièce jointe dans l'objet métier.

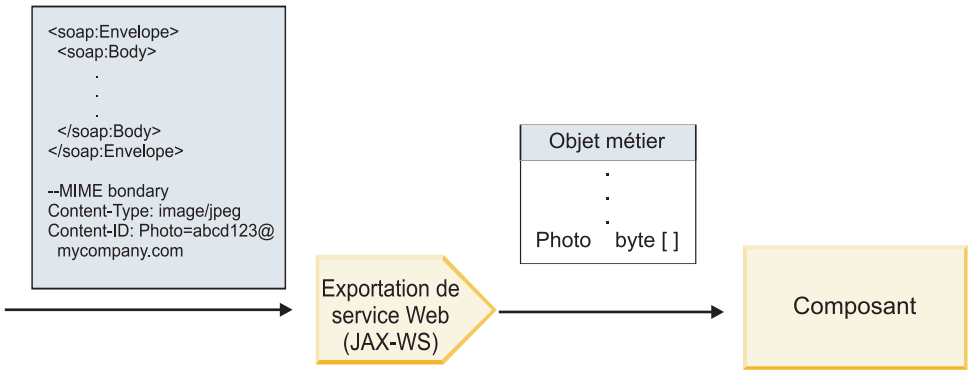


Figure 58. Mode de traitement d'un message SOAP avec une pièce jointe référencée par la liaison d'exportation de service Web (JAX-WS)

Attributs de pièce jointe MTOM

- MTOM peut prendre en charge les pièces jointes dans des structures imbriquées.
- MTOM est uniquement disponible pour le type base64Binary.
- MTOM peut prendre en charge les pièces jointes dans les structures imbriquées, cela signifiant que l'élément **bodyPath** associé aux pièces jointes MTOM est l'emplacement **xpath** de l'élément dans lequel la pièce jointe MTOM est contenue. La logique de calcul de **bodyPath** suit strictement le schéma permettant de générer l'emplacement **xpath** comme indiqué dans les exemples ci-après :
  - Pour un type autre que le tableau (**maxOccurs** a pour valeur 1) : /sendImage/input/imageData
  - Pour un type tableau (**maxOccurs** > 1) : /sendImage/input/imageData[1]
- Les types de pièces jointes mixtes ne sont pas pris en charge, cela signifiant que si MTOM est activé pour la liaison d'importation, la pièce jointe MTOM sera générée. Si MTOM est désactivé ou si la valeur de la configuration MTOM est la valeur par défaut pour la liaison d'exportation, le message MTOM entrant n'est pas pris en charge.

Pièces jointes référencées : éléments de type swaRef : Advanced

Vous pouvez envoyer et recevoir des messages SOAP qui incluent des pièces jointes représentées dans l'interface des services comme des éléments de type swaRef.

Un élément de type swaRef est défini dans Web Services Interoperability Organization (WS-I) *Attachments Profile* Version 1.0 (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), qui définit la manière dont les éléments de message sont associés aux composants MIME.

Dans le message SOAP, le corps du message SOAP contient un élément de type swaRef qui identifie l'ID contenu de la pièce jointe.

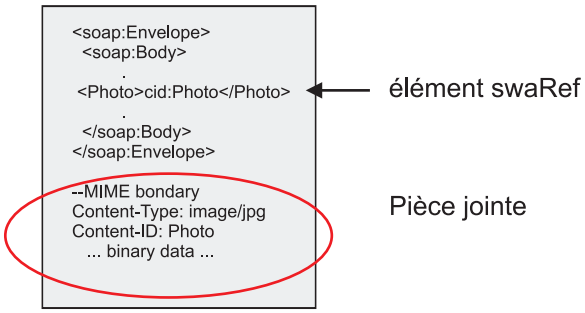


Figure 59. Message SOAP avec élément swaRef

Le WSDL de ce message SOAP contient un élément de type swaRef dans une portion de message qui identifie la pièce jointe.

```
<element name="sendPhoto">
  <complexType>
     <sequence>
       <element name="Photo" type="wsi:swaRef"/>
     </sequence>
  </complexType>
 </element>
```

Le WSDL doit également contenir une liaison MIME qui indique que les messages MIME composite doivent être utilisés.

Remarque : Le WSDL n'inclut *pas* de liaison MIME pour l'élément de message de type swaRef spécifique, car les liaisons MIME ne s'appliquent qu'aux portions de message de niveau supérieur.

La propagation des pièces jointes représentées comme des éléments de type swaRef n'est possible qu'entre composants de flux de médiation. Si un autre type de composant doit accéder à la pièce jointe ou servir de cible lors de la propagation de celle-ci, utilisez un composant de flux de médiation pour déplacer la pièce jointe vers un emplacement accessible par le composant.

Traitement entrant des pièces jointes

Utilisez Integration Designer pour configurer une liaison d'exportation afin de recevoir la pièce jointe. Vous créez un module, ainsi que son interface et ses opérations associées, et notamment un élément de type swaRef. Créez ensuite une liaison de service Web (JAX-WS).

Remarque : Pour obtenir des informations détaillées, reportez-vous à la rubrique «Utilisation des pièces jointes» du centre de documentation de Integration Designer.

Lorsqu'un client transmet un message SOAP avec une pièce jointe swaRef à un composant SCA (Service Component Architecture), la liaison d'exportation de service Web (JAX-WS) commence par supprimer cette pièce jointe. Elle analyse ensuite la partie SOAP du message et crée un objet métier. Enfin, la liaison définit l'ID contenu de la pièce jointe dans l'objet métier.

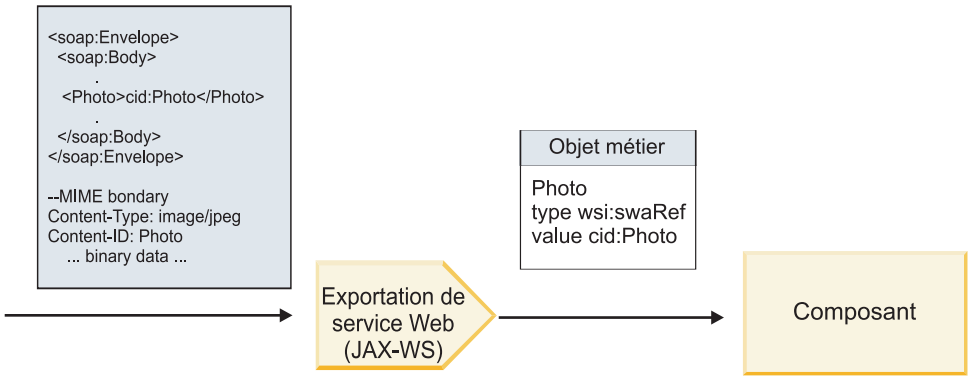


Figure 60. Méthode de traitement d'un message SOAP avec une pièce jointe swaRef par la liaison d'exportation de service Web (JAX-WS)

Accès aux métadonnées de pièce jointe dans un composant de flux de médiation

Comme illustré dans la figure 16, à la page 87, lorsque des composants accèdent à des pièces jointes swaRef, l'identificateur de contenu de la pièce jointe apparaît comme élément de type swaRef.

Chaque pièce jointe d'un message SOAP contient également un élément **attachments** correspondant dans l'objet SMO. Lorsqu'il utilise le type WS-I swaRef, l'élément **attachments** inclut le type de contenu de la pièce jointe et l'ID contenu, ainsi que les données binaires de la pièce jointe.

Pour obtenir la valeur d'une pièce jointe swaRef, il est donc nécessaire d'obtenir la valeur de l'élément de type swaRef, puis de rechercher l'élément **attachments** avec la valeur **contentID** correspondante. Notez que pour la valeur **contentID**, le préfixe **cid:** est généralement supprimé de la valeur swaRef.

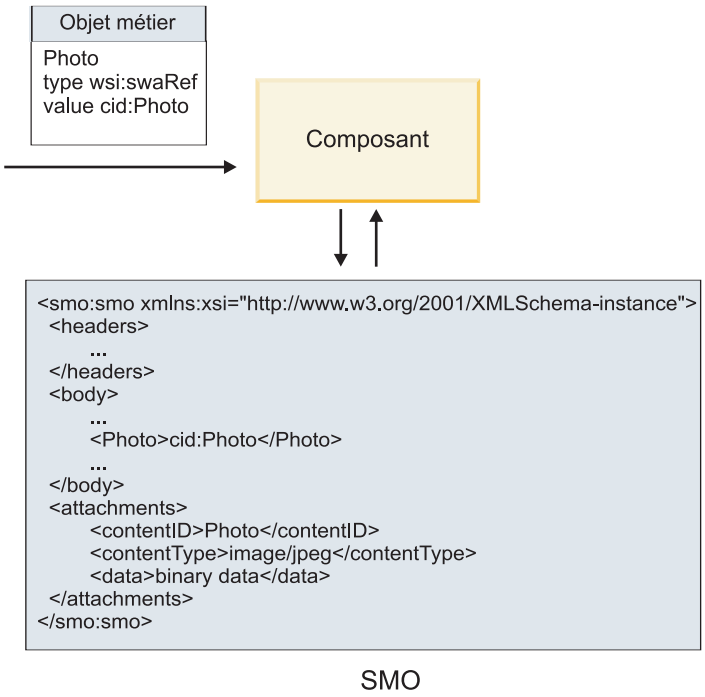


Figure 61. Affichage des pièces jointes swaRef dans l'objet SMO

Traitement sortant

Utilisez Integration Designer pour configurer une liaison d'importation de service Web (JAX-WS) de sorte qu'elle appelle un service Web externe. La liaison d'importation est configurée avec un document WSDL qui décrit le service Web à appeler et définit la pièce jointe qui sera transmise au service Web.

Lorsqu'un message SCA est reçu par une liaison d'importation de service Web (JAX-WS), les éléments de type swaRef sont envoyés comme pièces jointes si l'importation est connectée à un composant de flux de médiation et que l'élément de type swaRef possède un élément **attachments** correspondant.

Pour le traitement sortant, les éléments de type swaRef sont toujours envoyés avec leur ID contenu, mais le module de médiation doit s'assurer qu'il existe un élément **attachments** correspondant avec une valeur **contentID** correspondante.

Remarque : Pour se conformer au profil des pièces jointes WS-I, la valeur **content ID** doit suivre le "codage des portions content-id," comme décrit dans la section 3.8 de WS-I *Attachments Profile* 1.0.

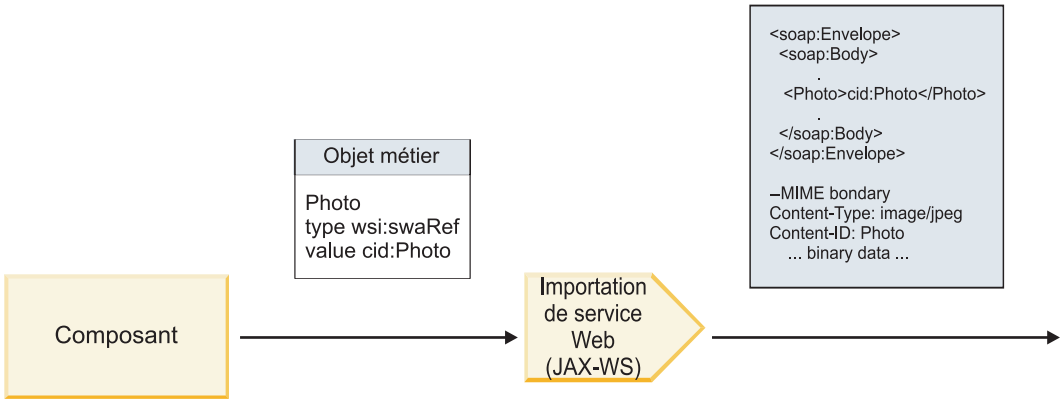


Figure 62. Méthode de génération d'un message SOAP avec une pièce jointe swaRef par la liaison d'importation de service Web (JAX-WS)

Définition des métadonnées de pièce jointe dans un composant de flux de médiation

Si, dans l'objet SMO, il existe une valeur d'élément de type swaRef et un élément **attachments**, la liaison prépare le message SOAP (avec la pièce jointe) et l'envoie à un destinataire.

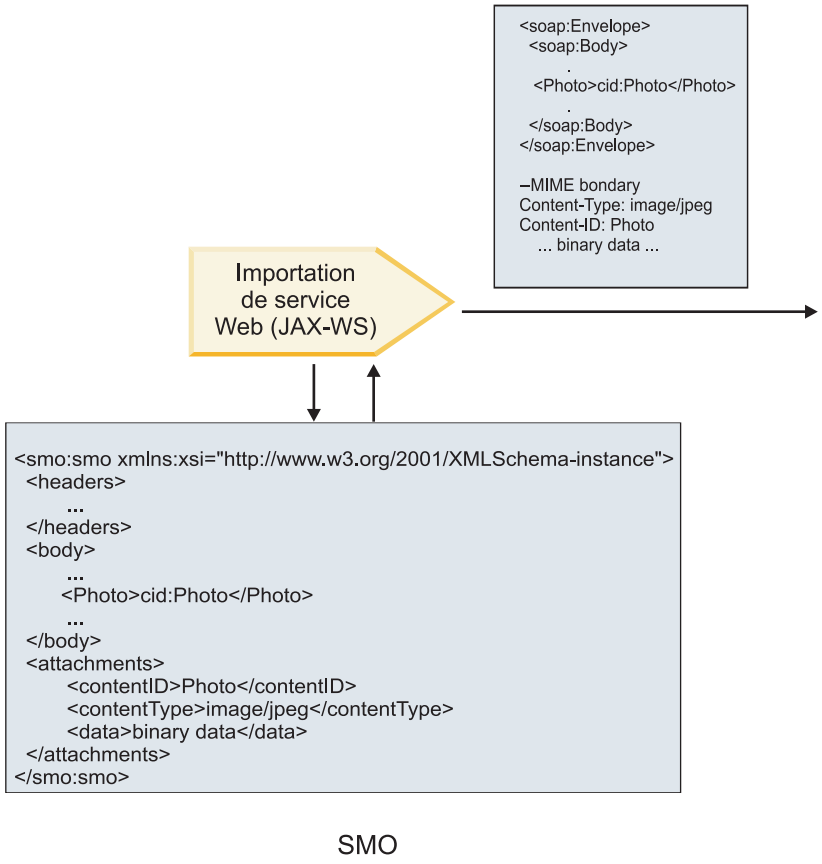


Figure 63. Méthode d'accès à une pièce jointe swaRef de l'objet SMO pour créer le message SOAP

L'élément **attachments** n'est présent dans l'objet SMO que si un composant de flux de médiation est connecté directement à l'importation ou l'exportation ; il n'est pas transmis entre les autres types de composant. Si les valeurs sont requises dans un module contenant d'autres types de composants, un composant de flux de médiation doit être utilisé pour copier ces valeurs dans un emplacement où elles

sont accessibles dans le module et un autre composant de flux de médiation doit être utilisé pour définir les valeurs correctes avant un appel sortant par l'intermédiaire d'une importation de service Web.

Important : Comme décrit dans la «représentation XML de l'objet SMO,» la primitive de médiation de mappage convertit les messages à l'aide d'une transformation XSLT 1.0. La transformation agit sur une sérialisation XML de l'objet SMO. La primitive de médiation de Mappage permet de spécifier la racine de la sérialisation et l'élément racine du document XML reflète cette racine.

Lorsque vous envoyez des messages SOAP comportant des pièces jointes, l'élément racine choisi détermine le mode de propagation des pièces jointes.

- Si vous utilisez «/body» comme racine de la mappe XML, toutes les pièces jointes sont propagées dans la mappe par défaut.
- Si vous utilisez «/» comme racine de la mappe, vous pouvez contrôler la propagation des pièces jointes.

Pièces jointes référencées : composants de message de niveau supérieur : Advanced

Vous pouvez envoyer et recevoir des messages SOAP qui incluent des pièces jointes binaires déclarées comme composants dans votre interface de service.

Dans un message SOAP à plusieurs parties MIME, le corps SOAP constitue la première partie du message et les pièces jointes sont dans les parties suivantes.

Quel est l'avantage d'envoyer ou de recevoir une pièce jointe référencée dans un message SOAP ? Les données binaires qui constituent la pièce jointe (dont la taille est souvent importante) sont conservées séparément du corps du message SOAP pour qu'elles n'aient pas besoin d'être analysées comme des données XML. Le traitement est donc plus efficace que si les données binaires étaient conservées dans un élément XML.

Types de messages SOAP avec pièces jointes référencées

A partir de la Version 7.0.0.3 de IBM Business Process Manager, vous avez le choix du mode de génération du message SOAP :

- **Messages compatibles WS-I**

  L'exécution peut générer des messages SOAP conformes à WS-I *Attachments Profile Version 1.0* et à WS-I *Basic Profile Version 1.1*. Dans un message SOAP conforme à ces profils, une seule partie du message est liée au corps SOAP ; pour celles liées en tant que pièces jointes, on utilise le codage des portions content-id (comme décrit dans WS-I *Attachments Profile Version 1.0*) pour lier la pièce jointe à la partie message.
- **Messages non compatibles WS-I**

  L'exécution peut générer des messages SOAP non conformes aux profils WS-I, mais qui sont compatibles avec les messages générés dans la Version 7.0 ou 7.0.0.2 de IBM Business Process Manager. Les messages SOAP utilisent des éléments de niveau supérieur nommés après la partie message avec un attribut **href** qui retient la pièce jointe **content-id**, mais le codage des portions content-id (comme décrit dans WS-I *Attachments Profile Version 1.0*) n'est pas utilisé.

Sélection de la compatibilité WS-I pour les exportations de services Web

Utilisez Integration Designer pour configurer une liaison d'exportation. Créez un module, ainsi que son interface et ses opérations associées. Créez ensuite une liaison de service Web (JAX-WS). La page Pièces jointes référencées affiche toutes les portions binaires de l'opération créée et vous sélectionnez les parties qui seront des pièces jointes. Indiquez ensuite, sur la page Specify the WS-I AP 1.0 compliance (Spécifier la compatibilité WS-I AP 1.0) de Integration Designer, l'un des choix suivants :

- **Use WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP compatible WS-I AP 1.0)**

Si vous sélectionnez cette option, vous indiquez aussi quelle partie du message doit être liée au corps SOAP.

Remarque : Cette option ne peut être utilisée que si le fichier WSDL correspondant est également compatible WS-I.

Un fichier WSDL généré par Integration Designer Version 7.0.0.3 est compatibles avec WS-I. Toutefois, vous ne pouvez pas sélectionner cette option, si vous importez un fichier WSDL qui n'est pas compatible avec WS-I.

- **Use non WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP non compatible WS-I AP 1.0)**

  Si vous sélectionnez cette option, qui est celle par défaut, la première partie du message est liée au corps SOAP.

Remarque : Seules les portions de message de niveau supérieur (les éléments définis dans le type de port WSDL (WSDL portType) comme portions du message en entrée ou en sortie) dont le type est binaire (base64Binary ou hexBinary) peuvent être envoyées ou reçues comme pièces jointes référencées.
Pour obtenir des informations détaillées, reportez-vous à la rubrique «Utilisation des pièces jointes» du centre de documentation de Integration Designer.

Pour des messages compatibles WS-I, le content-ID qui est généré dans le message SOAP est la concaténation des éléments suivants :

- La valeur de l'attribut **nom** de l'élément **wsdl:part** référencé par **mime:content**
- Le caractère **=**
- Une valeur globalement unique, comme un UUID (identificateur unique universel)
- Le caractère **@**
- Un nom de domaine valide

Traitement entrant des pièces jointes référencées

Lorsqu'un client transmet un message SOAP avec une pièce jointe à un composant SCA (Service Component Architecture), la liaison d'exportation de service Web (JAX-WS) commence par supprimer cette pièce jointe. Elle analyse ensuite la partie SOAP du message et crée un objet métier. Enfin, la liaison définit le code binaire de la pièce jointe dans l'objet métier.

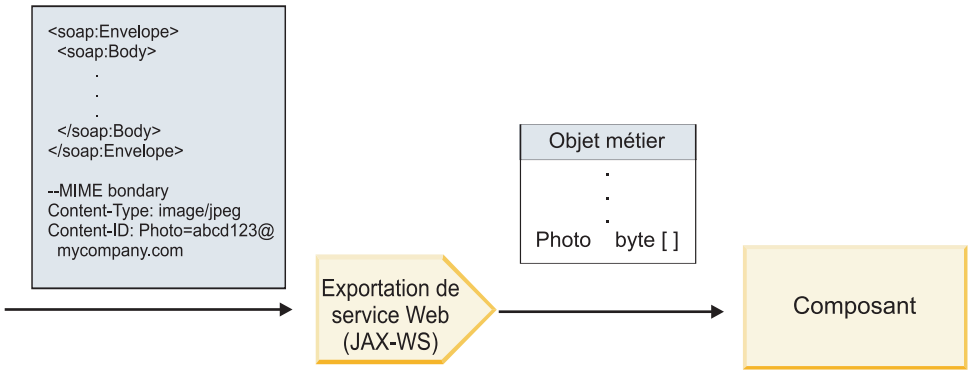


Figure 64. Mode de traitement d'un message SOAP compatible WS-I avec une pièce jointe référencée par la liaison d'exportation de service Web (JAX-WS)

Accès aux métadonnées de pièce jointe dans un composant de flux de médiation

Comme illustré dans la figure 19, à la page 90, lorsque des composants accèdent à des pièces jointes référencées, les données des pièces jointes apparaissent sous forme de tableau octal.

Chaque pièce jointe référencée d'un message SOAP contient également un élément **attachments** correspondant dans l'objet SMO. L'élément **attachments** inclut le type de contenu de la pièce jointe et le chemin d'accès à l'élément de corps de message dans lequel la pièce jointe est conservée.

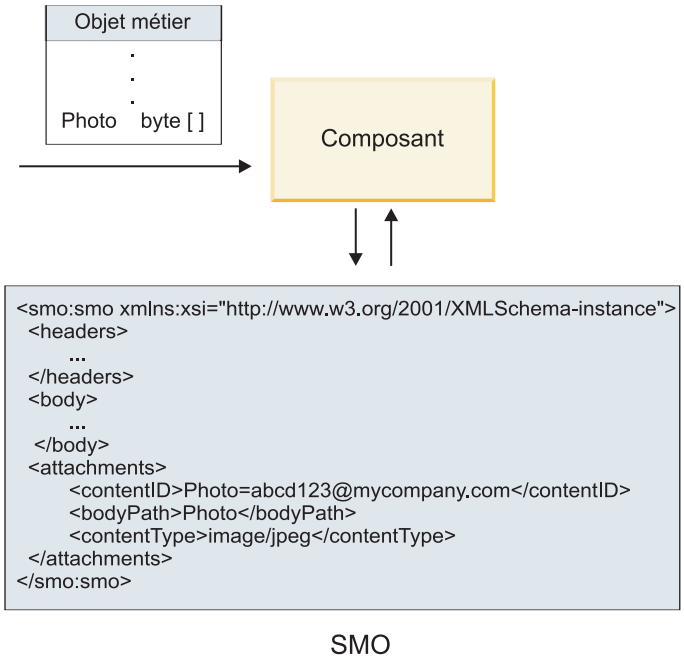


Figure 65. Affichage des pièces jointes référencées dans l'objet SMO

Important : Le chemin d'accès à l'élément de corps du message n'est pas automatiquement mis à jour si le message est converti et la pièce jointe, déplacée. Vous pouvez utiliser un flux de médiation pour mettre à jour l'élément **attachments** avec le nouveau chemin d'accès (par exemple, dans le cadre de la conversion ou à l'aide d'un configurateur d'élément de message distinct).

Mode de construction des messages SOAP sortant

Utilisez Integration Designer pour configurer une liaison d'importation de service Web (JAX-WS) de sorte qu'elle appelle un service Web externe. La liaison d'importation est configurée avec un document WSDL qui décrit le service Web à appeler et définit les portions du message qui doivent être transmises comme pièces jointes. Vous pouvez aussi préciser l'un des choix suivants sur la page Specify the WS-I AP 1.0 compliance (Spécifier la compatibilité WS-I AP 1.0) de Integration Designer :

- **Use WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP compatible WS-I AP 1.0)**

  Si vous sélectionnez cette option, indiquez aussi quelle partie du message doit être liée au corps SOAP ; toutes les autres sont liées à des pièces jointes ou à des en-têtes. Les messages envoyés par le lien n'inclut pas d'éléments du corps SOAP faisant référence aux pièces jointes ; la relation est exprimée au moyen de l'ID de contenu des pièces jointes incluant le nom de la partie du message.
- **Use non WS-I AP 1.0 compliant SOAP message (Utiliser un message SOAP non compatible WS-I AP 1.0)**

  Si vous sélectionnez cette option, qui est celle par défaut, la première partie du message est liée au corps SOAP ; toutes les autres sont liées aux pièces jointes et en-têtes. Les messages envoyés par la liaison incluent un ou plusieurs éléments du corps SOAP faisant référence aux pièces jointes au moyen d'un attribut **href**.

Remarque : La portion qui représente une pièce jointe, telle que définie dans le WSDL, doit être de type simple (base64Binary ou hexBinary). Si une portion est définie par un type complexe (complexType), celle-ci ne peut pas être liée comme pièce jointe.

Traitement sortant des pièces jointes référencées

La liaison d'importation utilise les informations dans le SMO pour déterminer comment les portions de messages binaires de niveau supérieur sont envoyées comme pièces jointes.

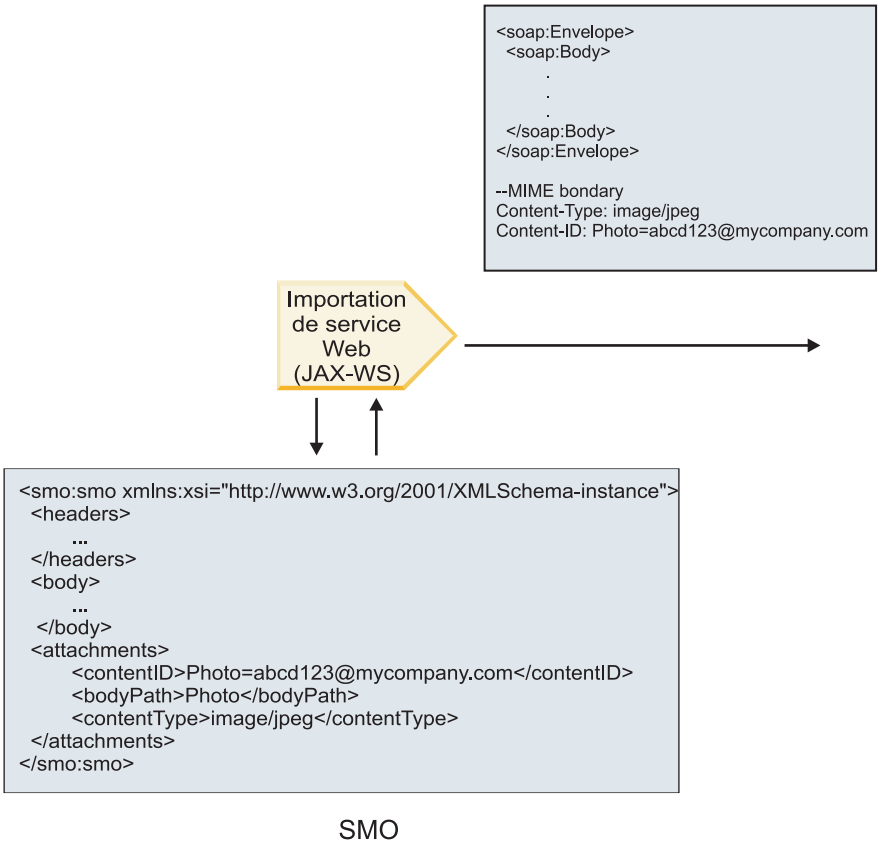


Figure 66. Mode d'accès à la pièce jointe référencée de l'objet SMO pour créer le message SOAP

L'élément **attachments** n'est présent dans l'objet SMO que si un composant de flux de médiation est connecté directement à l'importation ou l'exportation ; il n'est pas transmis entre les autres types de composant. Si les valeurs sont requises dans un module contenant d'autres types de composants, un composant de flux de médiation doit être utilisé pour copier ces valeurs dans un emplacement où elles sont accessibles dans le module et un autre composant de flux de médiation doit être utilisé pour définir les valeurs correctes avant un appel sortant par l'intermédiaire d'une importation de service Web.

La liaison utilise une combinaison des conditions suivantes pour déterminer comment (ou si) le message est envoyé :

- s'il existe une liaison MIME WSDL pour la portion du message binaire de niveau supérieur et, si tel est le cas, comment le type de contenu est défini,
- s'il existe un élément **attachments** dans le SMO dont la valeur **bodyPath** fait référence à une portion binaire de haut niveau,

Mode de création des pièces jointes quand un élément attachments se trouve dans le SMO.

Le tableau suivant montre comment une pièce jointe est créée et envoyée si le SMO contient un élément **attachments** avec un élément **bodyPath** correspondant une partie du nom du message :

Tableau 61. Mode de génération de la pièce jointe

Statut de la liaison MIME WSDL pour la portion de message binaire de haut niveau	**Mode de création et d'envoi du message**
Présent avec l'une des situations suivantes : • Aucun type de contenu défini pour la portion du message • Multiple types de contenu définis • Type de contenu avec caractère générique défini	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est définie à la valeur de l'élément attachments le cas échéant ; sinon, une valeur est générée. **Content-Type** est défini à la valeur de l'élément attachments le cas échéant ; sinon, il est défini à **application/octet-stream**.
Présent avec du contenu sans caractère générique pour la portion du message	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est définie à la valeur de l'élément attachments le cas échéant ; sinon, une valeur est générée. **Content-Type** est défini à la valeur de l'élément attachments le cas échéant ; sinon, il est défini au type défini dans l'élément de contenu MIME WSDL.
Non présent	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est définie à la valeur de l'élément attachments le cas échéant ; sinon, une valeur est générée. **Content-Type** est défini à la valeur de l'élément attachments le cas échéant ; sinon, il est défini à **application/octet-stream**. **Remarque :** L'envoi de portions de messages en tant que pièces jointes quand elles ne sont pas définies comme telles dans le langage WSDL peut empêcher la conformité avec WS-I Attachments Profile 1.0 et doit donc être si possible évité.

Mode de création des pièces jointes quand aucun élément attachments ne se trouve dans le SMO

Le tableau suivant montre comment une pièce jointe est créée et envoyée si le SMO ne contient pas d'élément **attachments** avec un élément **bodyPath** correspondant à une partie du nom du message :

Tableau 62. Mode de génération de la pièce jointe

Statut de la liaison MIME WSDL pour la portion de message binaire de haut niveau	**Mode de création et d'envoi du message**
Présent avec l'une des situations suivantes : • Aucun type de contenu défini pour la portion du message • Multiple types de contenu définis • Type de contenu avec caractère générique défini	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est généré. **Content-Type** est défini à **application/octet-stream**.

Tableau 62. Mode de génération de la pièce jointe (suite)

Statut de la liaison MIME WSDL pour la portion de message binaire de haut niveau	**Mode de création et d'envoi du message**
Présent avec du contenu sans caractère générique pour la portion du message	La portion du message est envoyée en tant que pièce jointe. **Content-Id** est généré. **Content-Type** est défini au type défini dans l'élément de contenu MIME WSDL.
Non présent	La portion du Message n'est pas envoyée en tant que pièce jointe.

Important : Comme décrit dans la «représentation XML de l'objet SMO,» la primitive de médiation de mappage convertit les messages à l'aide d'une transformation XSLT 1.0. La transformation agit sur une sérialisation XML de l'objet SMO. La primitive de médiation de Mappage permet de spécifier la racine de la sérialisation et l'élément racine du document XML reflète cette racine.

Lorsque vous envoyez des messages SOAP comportant des pièces jointes, l'élément racine choisi détermine le mode de propagation des pièces jointes.

- Si vous utilisez «/body» comme racine de la mappe XML, toutes les pièces jointes sont propagées dans la mappe par défaut.
- Si vous utilisez «/» comme racine de la mappe, vous pouvez contrôler la propagation des pièces jointes.

Pièces jointes non référencées : Advanced

Vous pouvez envoyer et recevoir des pièces jointes *non référencées* qui ne sont pas déclarées dans l'interface de service.

Dans ce type de message, le corps SOAP constitue la première partie et les pièces jointes figurent dans les parties suivantes. Aucune référence à la pièce jointe n'est incluse dans le corps SOAP.

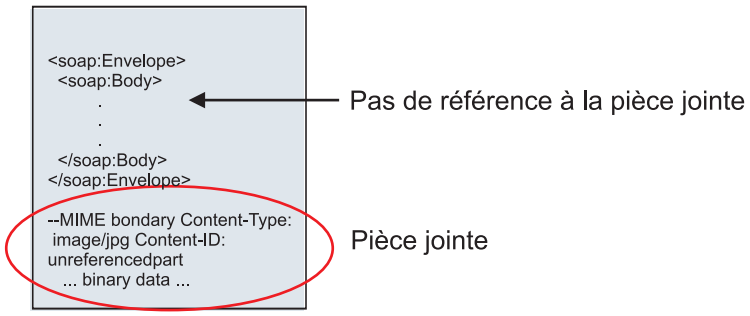


Figure 67. Message SOAP avec une pièce jointe non référencée

Vous pouvez envoyer un message SOAP avec une pièce jointe non référencée par une exportation de service Web vers une importation de service Web. Le message de sortie envoyé au service web cible contient la pièce jointe.

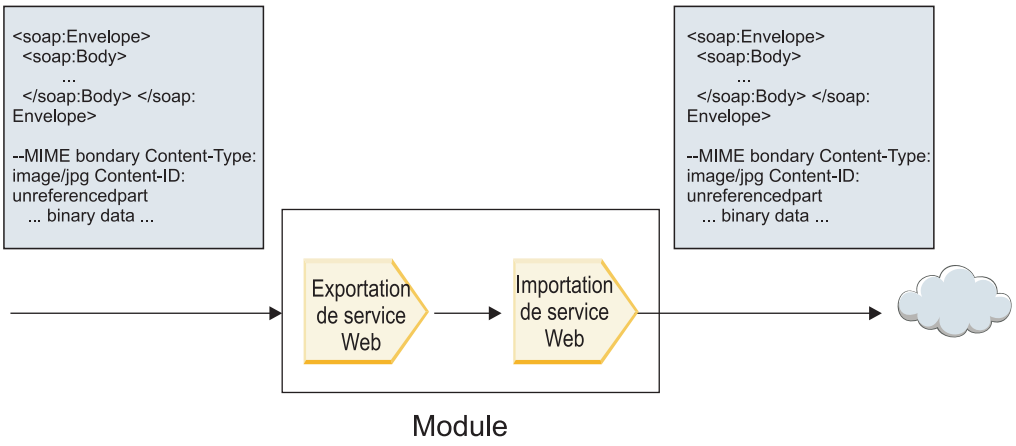


Figure 68. Pièce jointe transitant par un module SCA

Dans la figure 23, à la page 95, le message SOAP comprenant une pièce jointe transite par le module sans modification.

Vous pouvez également modifier le protocole SOAP à l'aide d'un composant de flux de médiation. Par exemple, vous pouvez utiliser le composant de flux de médiation pour extraire des données du message SOAP (dans ce cas, des données binaires dans le corps du message) et créer un message SOAP avec pièces jointes. Les données sont traitées comme partie de l'élément attachments d'un objet de message de service (SMO).

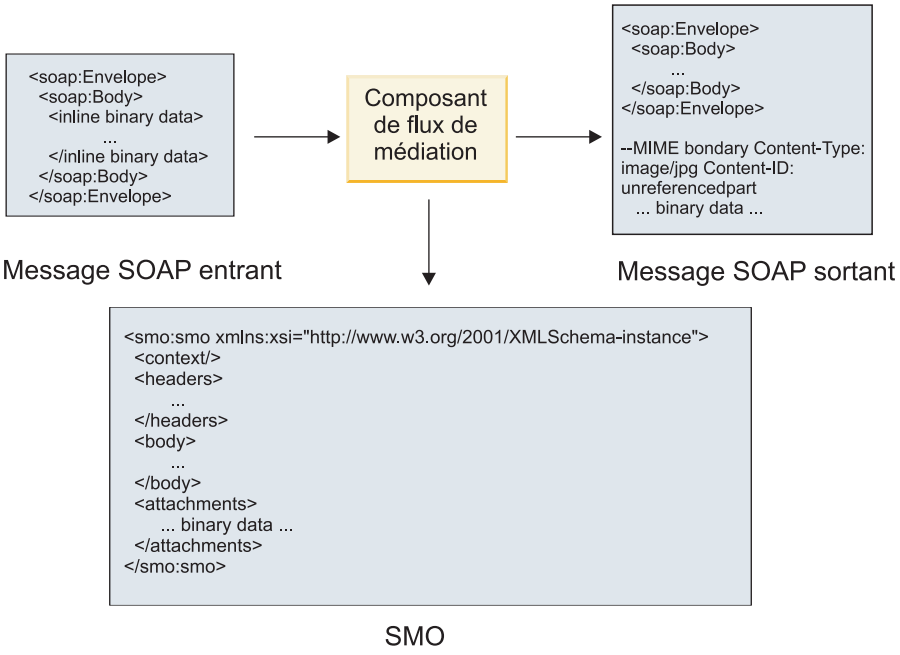


Figure 69. Message traité par un composant de flux de médiation

Inversement, le composant de flux de médiation peut transformer le message entrant en extrayant et en codant la pièce jointe, puis en transmettant le message sans pièce jointe.

Plutôt que d'extraire les données d'un message SOAP entrant pour former un message SOAP avec pièces jointes, vous pouvez obtenir les données de la pièce jointe à partir d'une source externe (par exemple, une base de données).

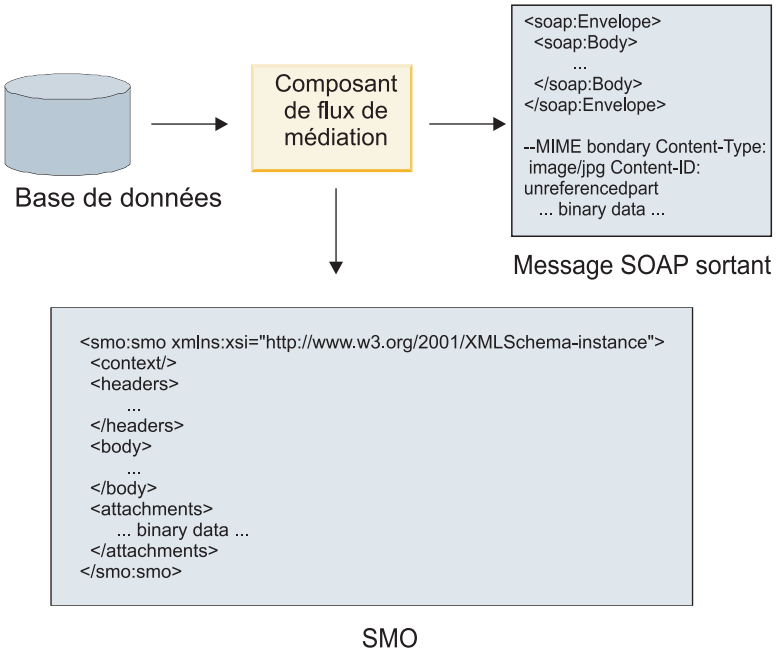


Figure 70. Pièce jointe obtenue d'une base de données et ajoutée au message SOAP

Inversement, le composant de flux de médiation peut extraire la pièce jointe d'un message SOAP entrant et traiter le message (par exemple, stocker la pièce jointe dans une base de données).

La propagation des pièces jointes non référencées n'est possible qu'entre composants de flux de médiation. Si un autre type de composant doit accéder à la pièce jointe ou servir de cible lors de la propagation de celle-ci, utilisez un composant de flux de médiation pour déplacer la pièce jointe vers un emplacement accessible par le composant.

Important : Comme décrit dans la «représentation XML de l'objet SMO,» la primitive de médiation de mappage convertit les messages à l'aide d'une transformation XSLT 1.0. La transformation agit sur une sérialisation XML de l'objet SMO. La primitive de médiation de Mappage permet de spécifier la racine de la sérialisation et l'élément racine du document XML reflète cette racine.

Lorsque vous envoyez des messages SOAP comportant des pièces jointes, l'élément racine choisi détermine le mode de propagation des pièces jointes.

- Si vous utilisez «/body» comme racine de la mappe XML, toutes les pièces jointes sont propagées dans la mappe par défaut.
- Si vous utilisez «/» comme racine de la mappe, vous pouvez contrôler la propagation des pièces jointes.

Utilisation d'une liaison de style Document WSDL avec des messages composites : Advanced

L'organisation WS-I (Web Services Interoperability Organization) a défini un ensemble de règles sur la manière dont les services web doivent être décrits par l'intermédiaire d'un WSDL et dont les messages SOAP correspondants doivent être constitués pour assurer leur interopérabilité.

Ces règles sont spécifiées dans WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). En particulier, WS-I Basic Profile 1.1 R2712 indique : "Une liaison document/littéral DOIT être mise en série comme une ENVELOPPE avec un soap:Body dont l'élément enfant est une instance de la déclaration de l'élément global référencé par la portion wsdl:message correspondante."

Cela signifie que si une liaison SOAP de style document est utilisée pour une opération dont les messages (entrée, sortie ou incident) sont définis avec plusieurs portions, une seule de ces portions doit être associée au corps du message SOAP pour la compatibilité à WS-I Basic Profile 1.1.

En outre, WS-I Attachments Profile 1.0 R2941 précise : "Un wsdl:binding d'une DESCRIPTION DOIT lier tous les wsdl:part d'un wsdl:message dans le wsdl:portType auquel il fait référence à l'un des suivants : soapbind:body, soapbind:header, soapbind:fault , soapbind:headerfault ou mime:content.".

Cela signifie que si une liaison SOAP de style document est utilisée pour une opération dont les messages (entrée, sortie ou incident) sont définis avec plusieurs portions, toutes les portions autres que celle sélectionnée pour être liée au corps SOAP doivent être liées en tant que pièces jointes ou en-têtes.

L'approche suivante est utilisée lorsque des descriptions WSDL sont générées pour les exportations avec des liaisons de service Web (JAX-WS et JAX-RPC) dans le cas suivant :

- Vous pouvez choisir quelle portion du message est liée au corps SOAP s'il existe plusieurs éléments typés non binaire. S'il n'existe qu'un seul élément typé non binaire, cet élément est lié automatiquement au corps SOAP.
- Pour la liaison JAX-WS, toutes les autres portions de message de type "hexBinary" ou "base64Binary" sont associées comme pièces jointes référencées. Voir «Pièces jointes référencées : composants de message de niveau supérieur», à la page 89.
- Toutes les autres portions de message sont associées comme en-têtes SOAP.

Les liaisons d'importation JAX-RPC et JAX-WS traitent la liaison SOAP d'un document WSDL existant avec des messages de style document en plusieurs parties, même si elle n'associe pas plusieurs portions au corps du message SOAP ; toutefois, vous ne pouvez pas générer de clients de service Web pour ces documents WSDL dans Rational Application Developer.

Remarque : La liaison JAX-RPC ne prend pas en charge les pièces jointes.

Le modèle recommandé lors de l'utilisation de messages composites avec une opération dont le style de document est une liaison SOAP est donc le suivant :

1. Usez le style encapsulé dans un document/littéral. Dans ce cas, les messages comportent toujours une seule partie, mais le référencement des messages doit être supprimé (comme décrit dans «Pièces jointes non référencées», à la page 94) ou être de type swaRef (comme décrit dans «Pièces jointes référencées : éléments de type swaRef», à la page 85).
2. Utilisé le style RPC/littéral. Dans ce cas, il n'existe aucune restriction sur la liaison WSDL en termes de nombre de portions associées au corps du message SOAP ; le message SOAP résultant possède toujours un enfant unique qui représente l'opération appelée, avec les portions de message correspondant aux enfants de cet élément.
3. Pour la liaison JAX-WS, il doit exister au plus une portion de message qui ne soit pas du "hexBinary" ou "base64Binary", sauf s'il est acceptable de lier les autres portions non binaires à des en-têtes SOAP.
4. Les autres cas ont le comportement décrit.

Remarque : Il existe des restrictions supplémentaires si vous utilisez des messages SOAP ne respectant pas le WS-I *Basic Profile Version 1.1*.

- La première portion du message doit être non binaire.
- Lorsqu'elle reçoit des messages SOAP de style document composite avec des pièces jointes référencées, la liaison JAX-WS s'attend à ce que chaque pièce jointe référencée soit représentée par un élément enfant de corps de message SOAP avec un attribut href dont la valeur identifie la pièce jointe d'après son ID contenu. La liaison JAX-WS envoie les pièces jointes référencées de tels messages de la même manière. Ce comportement n'est pas compatible avec WS-I Basic Profile.

Pour que vos messages soient compatibles avec Basic Profile, suivez l'approche 1, à la page 97 ou 2, à la page 97 de la liste précédente ou évitez d'utiliser des pièces jointes référencées pour de tels messages et utilisez à la place des pièces jointes non référencées ou de type swaRef.

Liaisons HTTP

La liaison HTTP permet de relier une architecture SOA à HTTP. Par conséquent, les applications HTTP existantes ou récemment développées peuvent être intégrées aux environnements d'architecture SOA (Service Oriented Architecture).

Le protocole HTTP (Hypertext Transfer Protocol) est largement utilisé pour le transfert d'informations sur le web. Lorsque vous travaillez avec une application externe qui utilise le protocole HTTP, une liaison HTTP est nécessaire. La liaison HTTP transforme les données entrantes en tant que message au format natif en un objet métier dans une application SCA. La liaison HTTP peut également transformer les données sortantes en tant qu'objet métier au format natif attendu par l'application externe.

Remarque : Si vous voulez interagir avec des clients et des services qui utilisent le protocole SOAP/HTTP de services Web, vous pouvez utiliser l'une des liaisons de service Web qui fournissent des fonctionnalités supplémentaires quant à la gestion des qualités de service standard des services Web.

La liste suivante répertorie certains scénarios courants pour l'utilisation de la liaison HTTP :

- Les services hébergés sur SCA peuvent appeler des applications HTTP en utilisant une importation HTTP.
- Les services hébergés sur SCA peuvent s'afficher eux-mêmes en tant qu'applications conformes à HTTP pour pouvoir être utilisés par des clients HTTP à l'aide d'une exportation HTTP.
- IBM Business Process Manager et Process Server peuvent communiquer entre eux via une infrastructure HTTP ; les utilisateurs peuvent donc gérer leurs communications en fonction de normes d'entreprise.
- IBM Business Process Manager et Process Server peuvent agir comme médiateurs des communications HTTP en tranformant et en routant les messages, ce qui améliore l'intégration des applications à l'aide d'un réseau HTTP.
- IBM Business Process Manager et Process Server peuvent être utilisés pour établir un pont entre HTTP et d'autres protocoles, comme les services Web SOAP/HTTP, les adaptateurs de ressources basés sur JCA (Java Connector Architecture), JMS, etc.

Pour plus d'informations sur la création de liaisons d'exportation et d'importation HTTP, consultez le centre de documentation de Integration Designer. Voir les rubriques**Développement des applications d'intégration** > **Accès aux services externes avec HTTP**>.

Présentation des liaisons HTTP :

La liaison HTTP offre une connectivité aux applications hébergées sur HTTP. Elle sert d'intermédiaire aux communications entre les applications HTTP et permet aux applications existantes basées sur HTTP d'être appelées depuis un module.

Liaisons d'importation HTTP

La liaison d'importation HTTP offre une connectivité sortante depuis les applications SCA (Service Component Architecture) vers des applications ou un serveur HTTP.

L'importation appelle une adresse URL de noeud final HTTP. L'URl peut être spécifiée de l'une des trois façons suivantes :

- L'URL peut être définie de manière dynamique dans les en-têtes HTTP par l'intermdiaire d'une URL de substitution dynamique.
- L'URL peut être définie de manière dynamique dans l'élément d'adresse cible SMO.

- L'URL peut être spécifiée en tant que propriété de configuration sur l'importation.

Cet appel est toujours synchrone par nature.

Bien que les appels HTTP soient toujours de type demande-réponse, l'importation HTTP prend en charge à la fois les opérations unidirectionnelles et bidirectionnelles et ignore la réponse dans le cas d'une opération unidirectionnelle.

Liaisons d'exportation HTTP

La liaison d'exportation HTTP offre une connectivité entrante depuis les applications HTTP vers une application SCA.

Une adresse URL est définie sur l'exportation HTTP. Les applications HTTP qui veulent envoyer des messages de demande à l'exportation utilisent cette adresse URL pour appeler l'exportation.

L'exportation HTTP prend également en charge les commandes ping.

Liaisons HTTP lors de l'exécution

Une importation avec une liaison HTTP au moment de l'exécution envoie une demande avec ou sans données dans le corps du message à partir de l'application SCA vers le service Web externe. Cette demande est émise à partir de l'application SCA vers le service Web externe, comme indiqué dans figure 26, à la page 99.

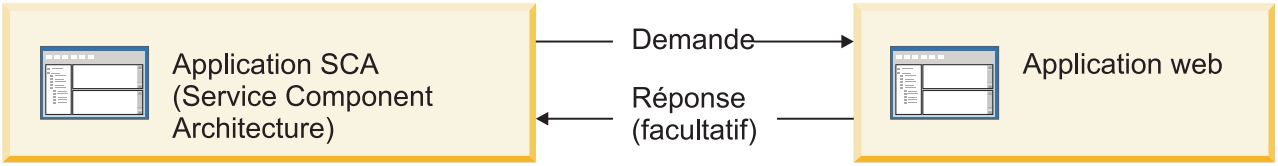


Figure 71. Flux d'une demande à partir de l'application SCA vers une application Web

L'importation avec la liaison HTTP peut éventuellement recevoir des données en retour issues d'une application Web dans la réponse à une demande.

Avec une exportation, la demande est faite par une application client à un service Web, comme indiqué dans figure 27, à la page 99.

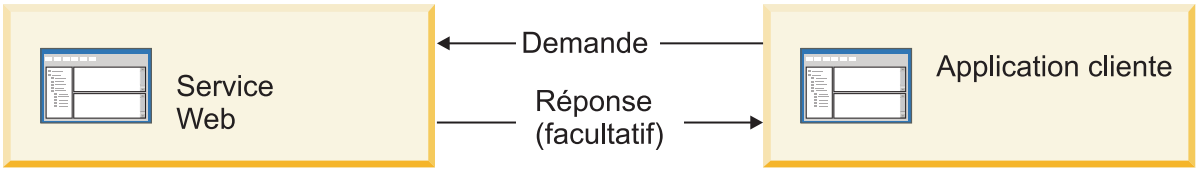


Figure 72. Flux d'une demande depuis l'application client vers le service Web.

Le service web est une application Web exécutée sur le serveur. L'exportation est implémentée dans cette application web en tant que servlet de telle sorte que le client envoie sa requête à une adresse URL. Le servlet transmet la requête à l'application SCA dans l'environnement d'exécution.

L'exportation peut éventuellement envoyer des données à l'application client en réponse à la requête.

En-têtes HTTP :

Les liaisons d'importation et d'exportation HTTP permettent d'effectuer la configuration des en-têtes HTTP et de leurs valeurs pour les messages sortants. L'importation HTTP utilise ces en-têtes pour les requêtes, tandis que l'exportation HTTP les exploite pour les réponses.

Les en-têtes et données de contrôle configurés de façon statique ont priorité sur les valeurs définies dynamiquement au moment de l'exécution. Toutefois, les valeurs de substitution dynamique d'adresse URL, de version et de méthode remplacement les valeurs statiques, qui sont dans les autres cas considérées comme valeurs par défaut.

La liaison prend en charge la nature dynamique de l'URL d'importation HTTP en déterminant la valeur des Méthode, Version et URL cible HTTP en phase d'exécution. Ces valeurs sont déterminées par l'extraction des valeurs de Référence de point de contact, URL de substitution dynamique, Version et Méthode.

- Pour Référence de point de contact, utilisez les API com.ibm.websphere.sca.addressing.EndpointReference ou définissez la zone /headers/SMOHeader/Target/address dans l'en-tête SMO.
- Pour URL de substitution dynamique, Version et Méthode, utilisez la section des paramètres de contrôle HTTP du message SCA (Service Component Architecture). Notez que l'URL de substitution dynamique est prioritaire par rapport à la référence de point de contact cible, mais que cette dernière s'appliquant entre les liaisons, elle est préférable et doit être utilisée chaque fois que possible.

Les données de contrôle et d'en-tête contenues dans les messages sortants sous les liaisons d'importation et d'exportation HTTP sont traitées dans l'ordre suivant :

1. Informations de contrôle et d'en-tête excluant les valeurs de méthode, de version et d'URL de substitution dynamique HTTP du message SCA (priorité la plus faible)
2. Les modifications effectuées au niveau de l'exportation/importation via la console d'administration
3. Les modifications effectuées au niveau de la méthode d'exportation ou d'importation via la console d'administration
4. Adresse cible spécifiée par l'intermédiaire de la référence de point de contact ou de l'en-tête SMO
5. URL de substitution dynamique, Version et Méthode du message SCA
6. Informations de contrôle et d'en-tête provenant du gestionnaire de données ou de la liaison de données (priorité supérieure)

L'importation et l'exportation HTTP ne rempliront les paramètres de contrôle et les en-têtes de direction sortants avec les données provenant du message entrant (HTTPExportRequest et HTTPImportResponse) que si la propagation de l'en-tête de protocole a la valeur **True**. Inversement, l'exportation et l'importation HTTP ne liront et ne traiteront les paramètres de contrôle et en-têtes sortants (HTTPExportResponse et HTTPImportRequest) que si la propagation de l'en-tête de protocole a la valeur **True**.

Remarque : Les modifications de la liaison de données ou du gestionnaire de données apportées aux en-têtes ou aux paramètres de contrôle dans la réponse d'importation ou la requête d'exportation n'altéreront pas les instructions de traitement du message au sein de la liaison d'imortation ou d'exportation et ne doivent être utilisées que pour diffuser les valeurs modifiées vers les composants SCA en aval.

Le service de contexte est chargé de la propagation du contexte (y compris les en-têtes de protocole comme l'en-tête HTTP et le contexte utilisateur comme l'ID de compte) tout au long d'un chemin d'appel SCA. Lors du développement dans IBM Integration Designer, vous pouvez contrôler la propagation du contexte par l'intermédiaire des propriétés d'importation et d'exportation. Pour plus de détails, reportez-vous aux informations relatives aux liaisons d'importation et d'exportation dans le centre de documentation IBM Integration Designer.

Structures d'en-tête HTTP fournies et prise en charge

Le tableau 34, à la page 101 spécifie en détail les demandes de requête et de réponse d'importation HTTP et d'exportation HTTP.

Tableau 63. Informations d'en-tête HTTP fournies

Nom du contrôle	Demande d'importation HTTP	Réponse d'importation HTTP	Demande d'exportation HTTP	Réponse d'exportation HTTP
URL	Ignored	Non définie	Lecture à partir du message de demande. **Remarque :** La chaîne de requête est également incluse dans le paramètre de contrôle de l'URL.	Ignored
Version (valeurs possibles : 1.0, 1.1 - Par défaut : 1.1)	Ignored	Non définie	Lecture à partir du message de demande	Ignored
Méthode	Ignored	Non définie	Lecture à partir du message de demande	Ignored
URL de substitution dynamique	Si elle est définie dans le gestionnaire de données ou la liaison de données, elle remplace l'URL d'importation HTTP. Inscrite sur le message dans la ligne de requête. **Remarque :** La chaîne de requête est également incluse dans le paramètre de contrôle de l'URL.	Non définie	Non définie	Ignored
Version de substitution dynamique	Si elle est définie, elle se substitue à la version d'importation HTTP. Elle est inscrite sur le message dans la ligne de requête.	Non définie	Non définie	Ignored
Méthode de substitution dynamique	Si elle est définie, elle se substitue à la méthode d'importation HTTP. Elle est inscrite sur le message dans la ligne de requête.	Non définie	Non définie	Ignored

Tableau 63. Informations d'en-tête HTTP fournies (suite)

Nom du contrôle	Demande d'importation HTTP	Réponse d'importation HTTP	Demande d'exportation HTTP	Réponse d'exportation HTTP
Type de support (ce paramètre de contrôle transport une partie de la valeur de l'en-tête HTTP de type de contenu.)	Si elle existe, elle est inscrite dans le message en tant que partie de l'en-tête de type de contenu. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par le gestionnaire de données ou la liaison de données.	Lecture à partir du message de réponse, en-tête Content-Type	Lecture à partir du message de demande, en-tête Content-Type	Si elle existe, elle est inscrite dans le message en tant que partie de l'en-tête Content-Type. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par le gestionnaire de données ou la liaison de données.
Jeu de caractères (par défaut : UTF-8)	Si elle existe, elle est inscrite dans le message en tant que partie de l'en-tête de type de contenu. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par la liaison de données.	Lecture à partir du message de réponse, en-tête Content-Type	Lecture à partir du message de demande, en-tête Content-Type	Pris en charge, inscrit dans le message en tant que partie de l'en-tête de type de contenu. **Remarque :** Cette valeur d'élément de contrôle doit être fournie par la liaison de données.
Codage de transfert (valeurs possibles : chunked, identity. Valeur par défaut: identity)	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage appliqué lors de la transformation du message.	Lecture à partir du message de réponse	Lecture à partir du message de demande	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage appliqué lors de la transformation du message.
Codage de contenu (valeurs possibles : gzip, x-gzip, deflate, identity. Valeur par défaut: identity)	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage de la charge.	Lecture à partir du message de réponse	Lecture à partir du message de demande	Si elle existe, elle est inscrite dans le message sous forme d'en-tête et contrôle le mode d'encodage de la charge.
Longueur de contenu	Ignorée	Lecture à partir du message de réponse	Lecture à partir du message de demande	Ignorée
StatusCode (par défaut : 200)	Non prise en charge.	Lecture à partir du message de réponse	Non prise en charge.	Si elle existe, elle est inscrite dans le message dans la ligne de réponse
ReasonPhrase (par défaut : OK)	Non prise en charge.	Lecture à partir du message de réponse	Non prise en charge.	Valeur de contrôle ignorée. La valeur contenue sur la ligne de réponse du message est générée à partir de StatusCode.

Tableau 63. Informations d'en-tête HTTP fournies (suite)

Nom du contrôle	Demande d'importation HTTP	Réponse d'importation HTTP	Demande d'exportation HTTP	Réponse d'exportation HTTP
Authentification (propriétés multiples)	Si elle existe, elle est utilisée pour établir l'en-tête d'authentification de base. **Remarque :** La valeur de cet en-tête est uniquement encodée dans le protocole HTTP. Dans l'architecture SCA, cette donnée est décodée et transmise sous forme de texte en clair.	Non disponible	Lecture à partir de l'en-tête d'authentification de base du message de demande. La présence de cet en-tête n'indique pas que l'utilisateur a été authentifié. Il convient de contrôler l'authentification via la configuration du servlet. **Remarque :** La valeur de cet en-tête est uniquement encodée dans le protocole HTTP. Dans l'architecture SCA, cette donnée est décodée et transmise sous forme de texte en clair.	Non disponible
Proxy (contient des propriétés multiples : Host, Port, Authentication)	Si elle existe, elle permet d'établir la connexion via le serveur Proxy.	Non disponible	Non disponible	Non disponible
SSL (contient des propriétés multiples : Keystore, Keystore Password, Trustore, Trustore Password, ClientAuth)	Si elle est remplie et que L'URL de destination est HTTPS, elle est utilisée pour établir une connexion via SSL.	Non disponible	Non disponible	Non disponible

Liaisons de données HTTP :

Pour chaque mappage de données distinct entre un message SCA (Service Component Architecture) et un message de protocole HTTP, un gestionnaire de données ou une liaison de données HTTP doit être configuré. Les gestionnaires de données, qui fournissent une interface indépendante des liaisons pouvant être réutilisée avec différentes liaisons de transport, représentent l'approche recommandée ; les liaisons de données sont spécifiques à une liaison de transport particulière. Des classes de liaisons de données propres à HTTP sont fournies ; vous pouvez également créer des liaisons ou des gestionnaires de données personnalisés.

Remarque : Les trois classes de liaison de données HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML et HTTPServiceGatewayDataBinding) sont dépréciées à partir de IBM Business Process Manager, version 7.0. Au lieu d'utiliser les liaisons de données décrites dans cette rubrique, prenez en compte les gestionnaires de données suivants :

- Utilisez SOAPDataHandler au lieu de HTTPStreamDataBindingSOAP.
- Utilisez UTF8XMLDataHandler au lieu de HTTPStreamDataBindingXML
- Utilisez GatewayTextDataHandler au lieu de HTTPServiceGatewayDataBinding

Les liaisons de données suivantes sont disponibles pour les importations et les exportations HTTP : liaison de données binaire, liaison de données XML et liaison de données SOAP. Une liaison de données de réponse n'est pas nécessaire pour les opérations unidirectionnelles. Une liaison de données est représentée par le nom d'une classe Java dont les instances permettent à la fois les conversions de HTTP vers ServiceDataObject et vice versa. Un sélecteur de fonction doit être utilisé lors d'une exportation pour déterminer, conjointement avec les liaisons de méthodes, la liaison de données utilisée et l'opération appelée. Les liaisons de données fournies sont les suivantes :

- Les liaisons de données binaires qui traitent le corps des messages comme des données binaires non structurées. L'implémentation du schéma XSD de liaisons de données binaires se présente comme suit :

```
<xsd:schema elementFormDefault="qualified"
  targetNamespace="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:tns="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:complexType name="HTTPBaseBody">
    <xsd:sequence/>
  </xsd:complexType>

  <xsd:complexType name="HTTPBytesBody">
    <xsd:complexContent>
      <xsd:extension base="tns:HTTPBaseBody">
        <xsd:sequence>
          <xsd:element name="value" type="xsd:hexBinary"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

- Les liaisons de données XML qui prennent en charge le corps des messages en tant que données XML. L'implémentation des liaisons de données XML est similaire à celle des liaisons de données XML JMS et n'impose aucune restriction quant au schéma d'interface.
- Les liaisons de données SOAP qui prennent en charge le corps des messages en tant que données SOAP. L'implémentation des liaisons de données SOAP n'impose aucune restriction quant au schéma d'interface.

Implémentation des liaisons de données HTTP personnalisées

Cette section indique comment implémenter une liaison de données HTTP personnalisée.

Remarque : L'approche recommandée consiste à implémenter un gestionnaire de données personnalisé car il peut être réutilisé avec différentes liaisons de transport.

HTTPStreamDataBinding est la principale interface de gestion des messages HTTP personnalisés. Cette interface est conçue pour permettre le traitement de charges utiles importantes. Toutefois, pour que ce type d'implémentation fonctionne, cette liaison de données doit renvoyer les informations de contrôle et les en-têtes avant que le message ne soit inséré dans le flux.

Les méthodes et leur ordre d'exécution (indiqué ci-dessous) doivent être implémentés par la liaison de données personnalisée.

Pour personnaliser une liaison de données, créez une classe implémentant HTTPStreamDataBinding. La liaison de données doit comporter les quatre propriétés privées suivantes :

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders
- private yourNativeDataType nativeData

La liaison HTTP appellera la liaison de données personnalisée dans l'ordre suivant :

- Traitement sortant (objet de données vers format natif) :
  1. setDataObject(...)
  2. setHeaders(...)
  3. setControlParameters(...)
  4. setBusinessException(...)
  5. convertToNativeData()
  6. getControlParameters()
  7. getHeaders()
  8. write(...)
- Traitement entrant (format natif vers objet de données) :
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Vous devez appeler setDataObject(...) dans convertFromNativeData(...) pour définir la valeur de dataObject, qui est convertie du format de données natif en propriété privée "pDataObject".

```
public void setDataObject(DataObject dataObject)
   throws DataBindingException {
  pDataObject = dataObject;

}
public void setControlParameters(HTTPControl arg0) {
  this.pCtrl = arg0;
}

public void setHeaders(HTTPHeaders arg0) {
  this.pHeaders = arg0;
}
/*
* Ajouter l'en-tête http "IsBusinessException" dans pHeaders.
* Deux étapes :
* 1. Supprimer d'abord tous les en-têtes portant le nom IsBusinessException (insensible à la casse).
*   Cette opération garantit qu'un seul en-tête est présent.
* 2. Ajouter le nouvel en-tête "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //Supprimer d'abord tous les en-têtes portant le nom IsBusinessException (insensible à la casse).
  //Cette opération garantit qu'un seul en-tête est présent.
  //Ajouter le nouvel en-tête "IsBusinessException", exemple de code :
  HTTPHeader header=HeadersFactory.eINSTANCE.createHTTPHeader();
  header.setName("IsBusinessException");
  header.setValue(Boolean.toString(isBusinessException));
  this.pHeaders.getHeader().add(header);

}
public HTTPControl getControlParameters() {
  return pCtrl;
}
public HTTPHeaders getHeaders() {
  return pHeaders;
}
public DataObject getDataObject() throws DataBindingException {
  return pDataObject;
}
```

```
/*
* Extraire l'en-tête "IsBusinessException" de pHeaders, renvoyer sa valeur booléenne
*/
public boolean isBusinessException() {
  String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
  boolean result=Boolean.parseBoolean(headerValue);
  return result;
}
public void convertToNativeData() throws DataBindingException {
  DataObject dataObject = getDataObject();
  this.nativeData=realConvertWorkFromSDOToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
  //Méthode développée par le client pour
  //Lire des données de HTTPInputStream
  //Les convertir en objet de données (DataObject)
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

Liaisons EJB

Les liaisons d'importation EJB (Enterprise JavaBeans permettent aux composants SCA d'appeler les services fournis par la logique métier de Java EE exécutée sur un serveur Java EE. Les liaisons d'exportation EJB permettent aux composants d'être affichés comme des EJB (Enterprise JavaBeans) pour que la logique métier de Java EE puisse appeler les composants SCA qui ne seraient autrement pas disponibles.

Liaisons d'importation EJB :

Les liaisons d'importation EJB permettent à un module SCA d'appeler des implémentations EJB en indiquant la manière dont le module utilisateur est lié à l'EJB externe. L'importation de services à partir d'une implémentation EJB externe permet aux utilisateurs de relier leur logique métier à l'environnement IBM Business Process Manager et de participer à un processus métier.

Vous pouvez utiliser Integration Designer pour créer des liaisons d'importation EJB. Vous pouvez utiliser l'une des procédures suivantes pour générer les liaisons :

- Création d'une importation EJB à l'aide de l'assistant de service externe

  Vous pouvez utiliser l'assistant de service externe de Integration Designer pour générer une importation EJB à partir d'une implémentation existante. L'assistant de service externe crée des services en fonction des critères que vous fournissez. Il génère ensuite des objets métier, des interfaces et des fichiers d'importation à partir des services détectés.
- Création d'une importation EJB à l'aide de l'éditeur d'assemblage

  Vous pouvez créer une importation EJB dans un diagramme d'assemblage à l'aide de l'éditeur d'assemblage de Integration Designer. A partir de la palette, vous pouvez utiliser une importation ou utiliser une classe Java pour créer la liaison EJB.

L'importation générée contient des liaisons de données qui établissent la connexion Java-WSDL, ce qui évite le recours à un composant de passerelle Java. Vous pouvez connecter directement un composant avec une référence WSDL (Web Services Description Language) à l'importation EJB qui communique avec un service EJB à l'aide d'une interface Java.

L'importation EJB peut interagir avec la logique métier Java EE à l'aide du modèle de programmation EJB 2.1 ou du modèle de programmation EJB 3.0.

L'appel de la logique métier Java EE peut être local (uniquement pour EJB 3.0) ou éloigné.

- L'appel local est utilisé lorsque vous souhaitez appeler la logique métier Java EE qui se trouve sur le même serveur que l'importation.

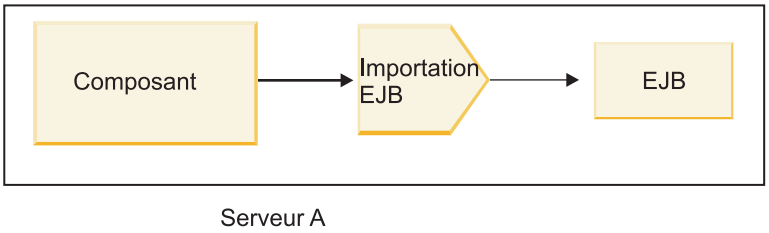


Figure 73. Appel en local d'un EJB (EJB 3.0 uniquement)

- L'appel éloigné est utilisé lorsque vous souhaitez appeler la logique métier Java EE qui ne se trouve pas sur le même serveur que l'importation.

  Par exemple, dans la figure ci-après, une importation EJB utilise RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) pour appeler une méthode EJB sur un autre serveur.

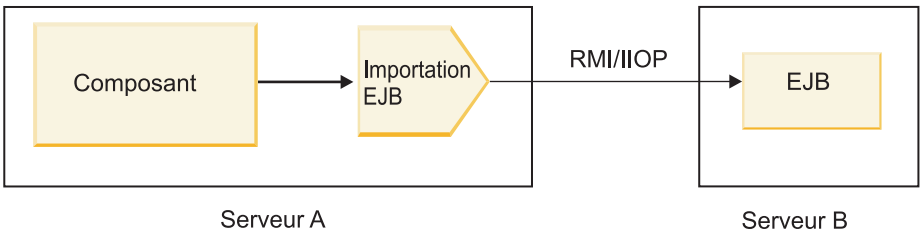


Figure 74. Appel éloigné d'un EJB

Lorsqu'il configure la liaison EJB, Integration Designer utilise le nom JNDI pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné).

Les liaisons d'importation EJB contiennent les principaux composants suivants :
- Gestionnaire de données JAX-WS
- Sélecteur d'erreurs EJB
- Sélecteur de fonction d'importation EJB

Si votre scénario utilisateur ne repose pas sur le mappage JAX-WS, vous risquez d'avoir besoin d'un gestionnaire de données, d'un sélecteur de fonction et d'un sélecteur d'erreur personnalisés pour accomplir les tâches habituellement dévolues aux composants des liaisons d'importation EJB. Ces tâches incluent le mappage normalement effectué par l'algorithme de mappage personnalisé.

Liaisons d'exportation EJB :

Les applications Java EE externes peuvent appeler un composant SCA par l'intermédiaire d'une liaison d'exportation EJB. L'utilisation d'une exportation EJB permet d'exposer des composants SCA pour que les applications Java EE externes puissent appeler ces composants à l'aide du modèle de programmation des EJB.

Remarque : L'exportation EJB est un bean sans état.

Vous pouvez utiliser Integration Designer pour créer des liaisons EJB. Vous pouvez utiliser l'une des procédures suivantes pour générer les liaisons :
- Création de liaisons d'exportation EJB à l'aide de l'assistant de service externe

Vous pouvez utiliser l'assistant de service externe de Integration Designer pour générer un service d'exportation EJB à partir d'une implémentation existante. L'assistant de service externe crée des services en fonction des critères que vous fournissez. Il génère ensuite des objets métier, des interfaces et des fichiers d'exportation à partir des services détectés.

- Création de liaisons d'exportation EJB à l'aide de l'éditeur d'assemblage.

  Vous pouvez créer une exportation EJB à l'aide de l'éditeur d'assemblage de Integration Designer.

Important : Un client J2SE (Java 2 Platform, Standard Edition) ne peut pas appeler le client d'exportation EJB qui est généré dans Integration Designer.

Vous pouvez générer la liaison à partir d'un composant SCA existant ou générer une exportation avec une liaison EJB pour une interface Java.

- Lorsque vous générez une exportation pour un composant SCA existant qui possède une interface WSDL, une interface Java est affectée à l'exportation.
- Lorsque vous générez une exportation pour une interface Java, vous pouvez sélectionner un WSDL ou une interface Java pour l'exportation.

  Remarque : Une interface Java utilisée pour créer une exportation EJB est soumise aux limitations suivantes en ce qui concerne les objets (exceptions et paramètres en entrée et en sortie) transmis comme paramètres sur un appel éloigné :
  - Ils doivent être de type concret (au lieu d'une interface ou d'un type abstrait).
  - Ils doivent respecter la spécification Enterprise JavaBeans. Ils doivent être sérialisables et contenir le constructeur no-argument par défaut et toutes les propriétés doivent être accessibles par l'intermédiaire de méthodes getter et setter.

    Pour obtenir des informations sur la spécification Enterprise JavaBeans, reportez-vous au site Web de Sun Microsystems, Inc. à l'adresse suivante : http://java.sun.com.

  En outre, l'exception doit correspondre à une exception vérifiée, héritée de java.lang.Exception et doit être unique (la génération de plusieurs exceptions de type vérifiée n'est pas prise en charge).

  Notez par ailleurs que l'interface métier d'un bean enterprise Java est une interface Java standard et ne doit pas étendre javax.ejb.EJBObject ou javax.ejb.EJBLocalObject. Les méthodes de l'interface métier ne doivent pas générer d'exception java.rmi.Remote.Exception.

Les liaisons d'exportation EJB peuvent interagir avec la logique métier Java EE à l'aide du modèle de programmation EJB 2.1 ou du modèle de programmation EJB 3.0.

L'appel peut être local (uniquement pour EJB 3.0) ou éloigné.

- L'appel local est utilisé lorsque la logique métier Java EE appelle un composant SCA qui se trouve sur le même serveur que l'exportation.
- L'appel éloigné est utilisé lorsque la logique métier Java EE se trouve pas sur le même serveur que l'exportation.

  Par exemple, dans la figure ci-après, un EJB utilise RMI/IIOP pour appeler un composant SCA sur un autre serveur.

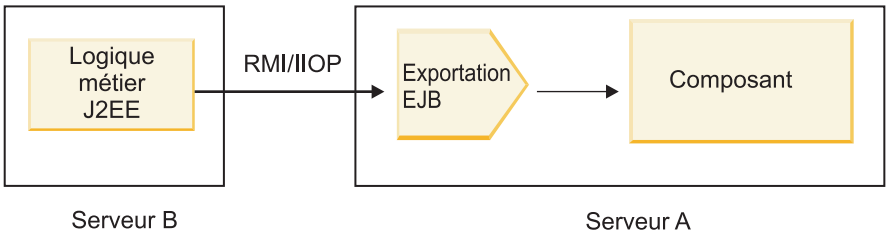


Figure 75. Appel éloigné d'un client vers un composant SCA par l'intermédiaire d'une exportation EJB

Lorsqu'il configure la liaison EJB, Integration Designer utilise le nom JNDI pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné).

Les liaisons d'exportation EJB contiennent les principaux composants suivants :

- Gestionnaire de données JAX-WS
- Sélecteur de fonction d'exportation EJB

Si votre scénario utilisateur ne repose pas sur le mappage JAX-WS, vous risquez d'avoir besoin d'un gestionnaire de données et d'un sélecteur de fonction personnalisés pour accomplir les tâches habituellement dévolues aux composants des liaisons d'exportation EJB. Ces tâches incluent le mappage normalement effectué par l'algorithme de mappage personnalisé.

Propriétés des liaisons EJB :

Les liaisons d'importation EJB utilisent leurs noms JNDI configurés pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné). Les liaisons d'importation et d'exportation EJB utilisent le gestionnaire de données JAX-WS pour la transformation des données. La liaison d'importation EJB utilise un sélecteur de fonction d'importation EJB et un sélecteur d'erreur EJB, tandis que la liaison d'exportation EJB utilise un sélecteur de fonction d'exportation EJB.

Noms JNDI et liaisons d'importation EJB :

Lorsqu'il configure la liaison EJB lors d'une importation, Integration Designer utilise le nom JNDI pour déterminer le niveau du modèle de programmation d'EJB et le type d'appel (local ou éloigné).

Si aucun nom JNDI n'est spécifié, la liaison d'interface EJB est utilisée. Les noms par défaut créés varient suivant que vous appelez EJB 2.1 JavaBeans ou EJB 3.0 JavaBeans.

Remarque : Pour plus de détails sur les conventions de d'affectation de nom, reportez-vous à la rubrique "Présentation des liaisons d'application EJB 3.0" dans le centre de documentation WebSphere Application Server.

- EJB 2.1 JavaBeans

  Le nom JNDI par défaut présélectionné par Integration Designer correspond à la liaison EJB 2.1 par défaut, qui prend la forme **ejb/** plus l'interface home, séparés par des barres obliques.

  Par exemple, pour l'interface home de JavaBeans EJB 2.1 de com.mycompany.myremotebusinesshome, la liaison par défaut est :

  ```
  ejb/com/mycompany/myremotebusinesshome
  ```

  Pour EJB 2.1, seul l'appel EJB éloigné est pris en charge.
- EJB 3.0 JavaBeans

  Le nom JNDI par défaut présélectionné par Integration Designer pour le JNDI local correspond au nom de classe complet de l'interface locale précédé d'**ejblocal:**. Par exemple, pour l'interface complète de l'interface com.mycompany.mylocalbusiness locale, le JNDI EJB 3.0 présélectionné est :

  ```
  ejblocal:com.mycompany.mylocalbusiness
  ```

Pour l'interface com.mycompany.myremotebusiness éloignée, le JNDI EJB 3.0 présélectionné correspond à l'interface complète :

```
com.mycompany.myremotebusiness
```

Les liaisons d'application par défaut EJB 3.0 sont décrites dans l'emplacement suivant : Présentation des liaisons d'application EJB 3.0.

Integration Designer utilise le nom "abrégé" comme emplacement JNDI par défaut des EJB à l'aide du modèle de programmation de la version 3.0.

Remarque : Si la référence JNDI déployée de l'EJB cible est différente de l'emplacement de la liaison JNDI par défaut car un mappage personnalisé a été utilisé ou configuré, le nom JNDI cible doit être correctement spécifié. Vous pouvez spécifier le nom dans Integration Designer avant le déploiement ou, pour la liaison d'importation, vous pouvez modifier le nom dans la console d'administration (après le déploiement), pour qu'il corresponde au nom JNDI de l'EJB cible.

Pour plus d'informations sur la création de liaisons EJB, reportez-vous à la section relative à l'utilisation des liaisons EJB dans le Integration Designer centre de documentation.

Gestionnaire de données JAX-WS :

La liaison d'exportation Enterprise JavaBeans (EJB) utilise le gestionnaire de données JAX-WS pour transformer les objets métier de demande en paramètres d'objet Java et la valeur de retour d'objet Java en objet métier de réponse. La liaison d'exportation EJB utilise le gestionnaire de données JAX-WS pour transformer les EJB de demande en objets métier de demande et l'objet métier de réponse en valeur renvoyée.

Ce gestionnaire de données mappe les données de l'interface indiquée par SCA à l'interface Java de l'EJB cible (et vice versa) à l'aide de la spécification JAX-WS (Java API for XML Web Services) et de la spécification JAXB (Java Architecture for XML Binding).

Remarque : Le support actuel est limité aux spécifications JAX-WS 2.1.1 et JAXB 2.1.3.

Le gestionnaire de données spécifié au niveau de la liaison EJB est utilisé pour traiter les demandes, les réponses, les incidents et les exceptions d'exécution.

Remarque : Pour les erreurs, un gestionnaire de données propre peut être spécifié pour chaque erreur en indiquant la propriété de configuration faultBindingType. Cette dernière remplace la valeur spécifiée au niveau de la liaison EJB.

Le gestionnaire de données JAX-WS est utilisé par défaut lorsque la liaison EJB dispose d'une interface WSDL. Ce gestionnaire de données ne peut pas être utilisé pour transformer un message SOAP représentant un appel JAX-WS en objet de données.

La liaison d'importation EJB utilise un gestionnaire de données pour transformer un objet de données en matrice d'objets Java (Object[]). Lors des communications sortantes, le traitement suivant est effectué :

1. La liaison EJB définit le type et l'élément attendus ainsi que le nom de la méthode ciblée dans BindingContext, afin qu'ils soient identiques à ceux spécifiés dans le fichier WSDL.
2. La liaison EJB appelle la méthode de transformation pour l'objet de données nécessitant une transformation des données.
3. Le gestionnaire de données renvoie un tableau Object[] représentant les paramètres de la méthode (listés dans l'ordre suivant lequel ils sont définis dans la méthode).
4. La liaison EJB utilise le tableau Object[] pour appeler la méthode sur l'interface EJB cible.

La liaison prépare également un tableau Object[] pour traiter la réponse à partir de l'appel EJB.

- Le premier élément dans l'objet[] correspond à la valeur renvoyée par l'appel de méthode Java.

- Les valeurs suivantes représentent les paramètres d'entrée de la méthode.

Ces dernières sont nécessaires pour prendre en charge les paramètres de type entrée/sortie et sortie.

Pour les paramètres de type sortie, les valeurs doivent être renvoyées dans l'objet de données de réponse.

Le gestionnaire de données traite et transforme les valeurs rencontrées dans l'objet[], puis renvoie une réponse à l'objet de données.

Le gestionnaire de données prend en charge xs:AnyType, xs:AnySimpleType, and xs:Any, ainsi que d'autres types de données XSD. Pour activer la prise en charge de xs:Any, utilisez **@XmlAnyElement (lax=true)** pour la propriété JavaBeans dans le code Java, comme illustré dans l'exemple suivant :

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Cela permet de transformer l'objet de propriété dans TestType en champ xs:any. La valeur de classe Java utilisée dans le champ xs:any doit inclure l'annotation **@XmlAnyElement**. Par exemple, si Adresse est la classe Java utilisée pour remplir la matrice d'objets, elle doit comporter l'annotation **@XmlRootElement**.

Remarque : Pour personnaliser le mappage du type XSD aux types Java définis par la spécification JAX-WS, modifiez les annotations JAXB selon les besoins de votre entreprise. Le gestionnaire de données JAX-WS prend en charge xs:any, xs:anyType et xs:anySimpleType.

Les restrictions suivantes s'appliquent au gestionnaire de données JAX-WS :

- Le gestionnaire de données ne prend pas en charge l'annotation **@WebParam** de l'attribut d'en-tête.
- L'espace de noms pour les fichiers de schéma des objet métiers (fichiers XSD) n'inclut pas le mappage par défaut du nom de package Java. L'annotation **@XMLSchema** dans package-info.java ne fonctionne pas non plus. La seule manière de créer un fichier XSD avec un espace de nom consiste à utiliser les annotations **@XmlType** et **@XmlRootElement**. **@XmlRootElement** définit l'espace de nom cible pour l'élément global dans les types de JavaBeans.
- L'assistant d'importation EJB ne crée pas de fichiers XSD pour les classes non associées. La version 2.0 ne prend pas en charge l'annotation **@XmlSeeAlso**. Par conséquent, si la classe enfant n'est pas directement référencée par la classe parente, aucun fichier XSD n'est créé. La solution à ce problème consiste à exécuter SchemaGen pour ces classes enfants.

  SchemaGen est un utilitaire de ligne de commande (situé dans le répertoire *Rép_base_installation_WPS/bin*) fourni pour créer des fichiers XSD pour un bean donné. Ces fichiers XSD doivent être manuellement copiés vers le module pour que la solution fonctionne.

Sélecteur d'erreur EJB :

Le sélecteur d'erreur EJB détermine si un appel EJB a engendré une erreur, une exception d'exécution ou une réponse correcte.

Si une erreur est détectée, le sélecteur d'erreurs EJB renvoie le nom de l'erreur native à l'environnement d'exécution de la liaison afin que le gestionnaire de données JAX-WS puisse convertir l'objet exception en objet métier d'erreur.

En cas de réponse positive (pas d'erreur), la liaison d'importation EJB assemble une matrice d'objets Java (Object[]) pour renvoyer les valeurs.

- Le premier élément dans l'objet[] correspond à la valeur renvoyée par l'appel de méthode Java.
- Les valeurs suivantes représentent les paramètres d'entrée de la méthode.

Ces dernières sont nécessaires pour prendre en charge les paramètres de type entrée/sortie et sortie.

Pour les scénarios d'exception, la liaison assemble un objet[] et le premier élément représente l'exception générée par la méthode.

Le sélecteur d'erreurs peut renvoyer l'une des valeurs suivantes :

Tableau 64. Valeurs renvoyées

Type	Valeur renvoyée	Description
Erreur	**ResponseType.FAULT**	Renvoyée si la matrice Object[] transmise contient un objet d'exception.
Exception d'exécution	**ResponseType.RUNTIME**	Renvoyée si l'objet d'exception ne correspond à aucun des types d'exception déclarés sur la méthode.
Réponse normale	**ResponseType.RESPONSE**	Renvoyée dans tous les autres cas.

Si le sélecteur d'erreurs renvoie la valeur **ResponseType.FAULT**, le nom d'erreur natif est renvoyé. Ce nom est utilisé par la liaison pour déterminer le nom d'erreur WSDL correspondant à partir du modèle et appeler le gestionnaire de données d'erreur approprié.

Sélecteur de fonction EJB :

Les liaisons EJB utilisent un sélecteur de fonction d'importation (pour le traitement sortant) un sélecteur de fonction d'exportation (pour le traitement entrant) afin de déterminer la méthode EJB à appeler.

Sélecteur de fonction d'importation

Pour le traitement sortant, le sélecteur de fonction d'importation détermine le type de méthode EJB en fonction du nom de l'opération appelée par le composant SCA connecté à l'importation EJB. Le sélecteur de fonction recherche l'annotation @WebMethod sur la classe Java annotée par JAS-WS et générée par Integration Designer pour déterminer le nom de l'opération cible associée.

- Si l'annotation @WebMethod est présente, le sélecteur de fonction utilise l'annotation @WebMethod pour déterminer le mappage correct de la méthode Java pour la méthode WSDL.
- Si l'annotation @WebMethod est manquante, le sélecteur de fonction suppose que le nom de la méthode Java est identique au nom de l'opération appelée.

Remarque : Ce sélecteur de fonction n'est valide que pour une interface de type WSDL sur une importation EJB et non pour une interface de type Java sur une importation EJB.

Le sélecteur de fonction renvoie un objet java.lang.reflect.Method qui représente la méthode de l'interface EJB.

Le sélecteur de fonction utilise une matrice d'objets Java (Object[]) pour conserver la réponse de la méthode cible. Le premier élément de la matrice Object[] est une méthode Java possédant le nom du WSDL et le second correspond à l'objet métier en entrée.

Sélecteur de fonction d'exportation

Pour le traitement entrant, le sélecteur de fonction d'exportation détermine la méthode cible à appeler à partir de la méthode Java.

Le sélecteur de fonction d'exportation mappe le nom de l'opération Java appelée par le client EJB au nom de l'opération dans l'interface du composant cible. Le nom de la méthode est renvoyé sous forme de chaîne et résolu par l'environnement d'exécution de SCA en fonction du type d'interface du composant cible.

Liaisons EIS

Les liaisons de système d'information d'entreprise (EIS) permettent la connectivité entre les composants SCA et un système EIS externe. Cette communication est accomplie à l'aide des exportations et des importations EIS qui prennent en charge les adaptateurs de ressources JCA 1.5 et les adaptateurs Websphere.

Vos composants SCA peuvent exiger le transfert de données vers ou à partir d'un EIS externe. Lorsque vous créez un module SCA exigeant une telle connectivité, vous incluez (outre le composant SCA) une importation ou exportation avec une liaison EIS pour communiquer avec un EIS externe spécifique.

Les adaptateurs de ressources dans IBM Integration Developer sont utilisés dans le contexte d'une importation ou d'une exportation. Vous développez une importation ou une exportation à l'aide de l'assistant de service externe puis, lors du développement vous incluez l'adaptateur de ressources. Une importation EIS permettant à votre application d'appeler un service sur un système EIS ou une exportation EIS permettant à une application sur un système EIS d'appeler un service développé dans IBM Integration Developer sont créées avec un adaptateur de ressources. Par exemple, vous pourriez créer une importation avec un adaptateur JD Edwards pour appeler un service sur le système JD Edwards.

Lorsque vous utilisez un assistant de service externe, les informations de liaison EIS sont créées pour vous. Vous pouvez également utiliser un autre outil, l'éditeur d'assemblage, pour ajouter ou modifier des informations de liaison. Pour plus d'informations, voir Accès aux services externes à l'aide des adaptateurs.

Une fois que le module SCA contenant la liaison EIS est déployé sur le serveur, vous pouvez utiliser la console d'administration pour afficher les informations relatives à la liaison ou pour la configurer.

Présentation des liaisons EIS :

Lorsqu'elle est utilisée avec un adaptateur de ressources JCA, la liaison EIS (Enterprise Information System) permet d'accéder à des services sur un système d'information d'entreprise ou de rendre vos services disponibles auprès de l'EIS.

L'exemple suivant illustre la manière dont un module SCA appelé ContactSyncModule synchronise les informations de contact entre un système Siebel et un système SAP.

1. Le composant SCA appelé ContactSync écoute (via une exportation d'application EIS appelée Siebel Contact) les modifications apportées aux contacts Siebel.
2. Le composant ContactSync lui-même utilise une application SAP (via une importation d'application EIS) afin de mettre à jour les informations de contact SAP en conséquence.

Comme les structures de données utilisées pour le stockage de contacts sont différentes dans les systèmes Siebel et SAP, le composant SCA ContactSync doit assurer le mappage.

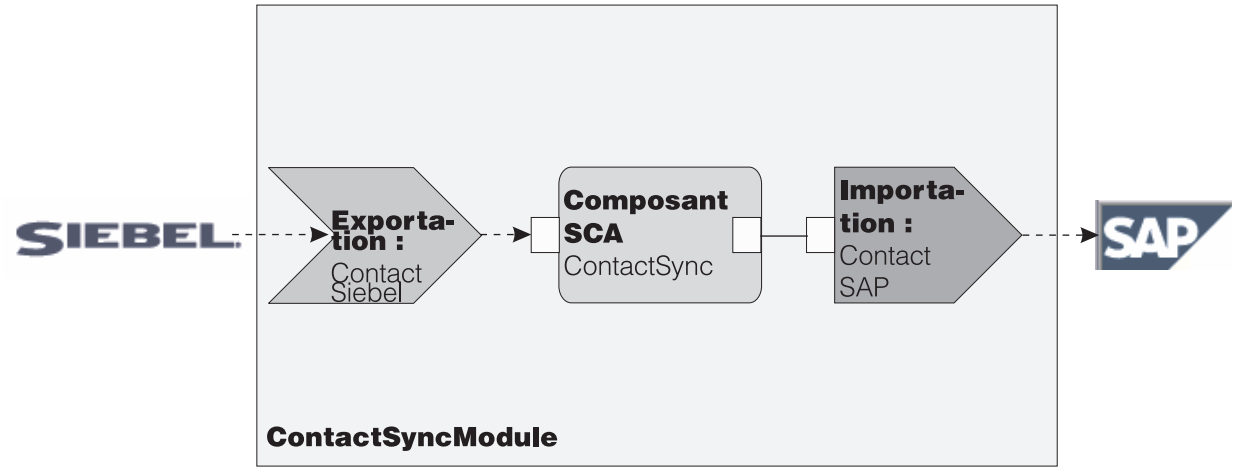


Figure 76. Flux provenant d'un système Siebel vers un système SAP

L'exportation Siebel Contact et l'importation SAP Contact ont les adaptateurs de ressources appropriés configurés.

Principales fonctionnalités des liaisons EIS :

Une importation EIS est une importation SCA (Service Component Architecture) qui permet aux composants du module SCA d'utiliser les applications EIS définies hors du module SCA. Une importation EIS permet de transférer des données d'un composant SCA vers un système EIS externe. Une exportation EIS permet de transférer des données d'un système EIS externe vers un module SCA.

Importations

La fonction de l'importation EIS est de combler l'écart entre les composants SCA et les systèmes EIS externes. Les applications externes peuvent être traitées comme des importations EIS. Dans ce cas, l'importation EIS envoie des données au système EIS externe et reçoit éventuellement des données en réponse.

L'importation EIS fournit aux composants SCA une vue uniforme des applications externes au module. Ceci permet aux composants de communiquer avec un système EIS externe, tel que SAP, Siebel ou PeopleSoft, via un modèle de programmation SCA homogène.

Côté client de l'importation, une interface est affichée par l'application d'importation EIS, laquelle comporte une ou plusieurs méthodes, chacune prenant des objets de données servant d'arguments et de valeurs de retour. Côté implémentation, une interface client commune (Common Client Interface - CCI) est implémentée par un adaptateur de ressources.

L'implémentation d'exécution d'une importation EIS connecte l'interface côté client et l'interface CCI. L'importation mappe l'appel de la méthode de l'interface à l'appel sur l'interface CCI.

Des liaisons sont créées à trois niveaux : une liaison d'interface, qui utilise les liaisons de méthode incluses, lesquelles utilisent les liaisons de données.

La liaison d'interface associe l'interface de l'importation à la connexion au système EIS qui fournit l'application. Ceci reflète le fait que l'ensemble d'applications, représenté par l'interface, est fourni par l'instance spécifique du système EIS et que la connexion permet l'accès à cette instance. L'élément de liaison contient des propriétés avec suffisamment d'informations pour créer la connexion (ces propriétés font partie de l'instance javax.resource.spi.ManagedConnectionFactory).

La liaison de méthode associe la méthode à l'interaction spécifique avec le système EIS. Pour JCA, cette interaction se caractérise par l'ensemble de propriétés de l'implémentation de l'interface javax.resource.cci.InteractionSpec. L'élément d'interaction de la liaison de méthode contient ces propriétés, ainsi que le nom de la classe, fournissant ainsi assez d'informations pour permettre l'interaction. La liaison de méthode utilise des liaisons de données décrivant le mappage de l'argument et du résultat de la méthode d'interface à la représentation EIS.

Le scénario d'exécution d'une importation EIS est le suivant :

1. La méthode de l'interface d'importation est appelée à l'aide du modèle de programmation SCA.
2. La demande, qui parvient au gestionnaire d'importation EIS, contient le nom de la méthode et ses arguments.
3. Tout d'abord, l'importation crée une implémentation de liaison d'interface. Ensuite, à partir des données de la liaisond'importation, il crée une ConnectionFactory et associe les deux. Autrement dit, l'importation appellesetConnectionFactory sur la liaison d'interface.
4. L'implémentation de liaison de méthode correspondant à la méthode appelée est créée.
5. L'instance javax.resource.cci.InteractionSpec est créée et remplie, puis les liaisons de données sont utilisées pour lier les arguments de la méthode à un format lisible par l'adaptateur de ressource.
6. L'interface CCI permet d'appliquer l'interaction.
7. Lors du retour de l'appel, la liaison de données permet de créer le résultat de l'appel et de renvoyer ce résultat au demandeur.

Exportations

La fonction de l'exportation EIS est de combler l'écart entre un composant SCA et un système EIS externe. Les applications externes peuvent être traitées comme des exportations EIS. Dans ce cas, l'application externe envoie ses données sous la forme de notifications périodiques. Une exportation EIS peut être considérée comme une application d'abonnement qui écoute une demande externe provenant d'un EIS. Le composant SCA qui utilise l'exportation EIS la voit comme une application locale.

L'exportation EIS fournit aux composants SCA une vue uniforme des applications externes au module. Ceci permet aux composants de communiquer avec un système EIS, tel que SAP, Siebel ou PeopleSoft, via un modèle de programmation SCA homogène.

L'exportation comporte une implémentation d'écouteur recevant des demandes du système EIS. L'écouteur implémente une interface d'écouteur spécifique à l'adaptateur de ressources. L'exportation contient également une interface d'implémentation de composant, exposée au système EIS via l'exportation.

L'implémentation d'exécution d'une exportation EIS connecte l'écouteur et l'interface d'implémentation de composant. L'exportation mappe la demande EIS à l'appel de l'opération appropriée sur le composant. Des liaisons sont créées à trois niveaux : une liaison d'écouteur, qui utilise une liaison de méthode native incluse, laquelle utilise une liaison de données.

La liaison d'écouteur associe l'écouteur qui reçoit des demandes au composant affiché via l'exportation. La définition d'exportation contient le nom du composant ; l'environnement d'exécution le localise et lui transmet des demandes.

La liaison de méthode native associe la méthode native ou le type d'événement reçu par l'écouteur à l'opération implémentée par le composant affiché au moyen de l'exportation. Il n'y a aucune relation entre la méthode appelée sur l'écouteur et le type d'événement ; tous les événements arrivent par l'intermédiaire d'une ou de plusieurs méthodes de l'écouteur. La liaison de méthode native utilise le sélecteur de fonction défini dans l'exportation pour extraire le nom de la méthode native des données entrantes et des liaisons de données pour lier le format de données du système EIS à un format lisible par le composant.

Le scénario d'exécution d'une exportation EIS est le suivant :

1. La demande EIS déclenche l'appel de la méthode sur l'implémentation de l'écouteur.
2. L'écouteur localise et appelle l'application d'exportation en lui transmettant tous les arguments d'appel.
3. L'exportation crée l'implémentation de liaison d'écouteur.
4. L'exportation instancie le sélecteur de fonction et le définit sur la liaison d'écouteur.
5. L'exportation initialise les liaisons de méthode native et les ajoute à la liaison de l'écouteur. Pour chaque liaison de méthode native, les liaisons de données sont également initialisées.
6. L'exportation appelle la liaison de l'écouteur.
7. Cette liaison localise les composants exportés et utilise le sélecteur de fonction pour extraire le nom de la méthode native.
8. Ce nom est utilisé pour localiser la liaison de méthode native, qui peut alors appeler le composant cible.

Le type d'interaction de l'adaptateur permet à la liaison d'exportation EIS d'appeler le composant cible de manière asynchrone (par défaut) ou synchrone.

Adaptateurs de ressources

Vous développez une importation ou une exportation à l'aide de l'assistant de service externe puis, lors du développement, vous incluez un adaptateur de ressources. Les adaptateurs livrés avec IBM Integration Designer pour accéder aux systèmes CICS, IMS, JD Edwards, PeopleSoft, SAP et Siebel sont réservés à des fins de développement et de test uniquement. Autrement dit, vous les utilisez pour développer et tester vos applications.

Après le déploiement de votre application, vous avez besoin d'adaptateurs d'exécution sous licence pour exécuter votre application. Toutefois, lorsque vous créez votre service, vous pouvez y intégrer l'adaptateur. La licence pour votre adaptateur peut vous permettre d'utiliser l'adaptateur intégré comme adaptateur d'exécution sous licence. Ces adaptateurs sont compatibles avec la norme JCA 1.5 (Java EE Connector Architecture). JCA, standard ouvert, est la norme Java EE pour la connectivité EIS. JCA fournit un cadre géré ; la qualité de service (QoS) est assurée par le serveur d'applications qui offre aux transactions la sécurité et la gestion du cycle de vie. Ces adaptateurs sont également compatibles avec la spécification Enterprise Metadata Discovery, à l'exception de l'adaptateur de ressources IBM CICS ECI et du connecteur IBM IMS pour Java.

Les adaptateurs WebSphere Business Integration Adapters, un ancien jeu d'adaptateurs, sont également pris en charge par l'assistant.

Ressources Java EE

Le module EIS, un module SCA qui repose sur le pattern des modules EIS, peut être déployé sur la plateforme Java EE.

Le déploiement d'un module EIS sur la plateforme Java EE génère une application prête à être exécutée, encapsulée dans un fichier EAR et déployée sur le serveur. Tous les artefacts et ressources Java EE sont créés ; l'application est configurée et prête pour l'exécution.

Propriétés dynamiques des spécifications d'interaction et de connexion JCA :

La liaison EIS peut accepter des données en entrée pour lesspécifications InteractionSpec et ConnectionSpec spécifiées, en utilisant un objet de données enfant bien défini qui accompagne la charge. Ceci permet des interactions demande-réponse dynamiques avec un adaptateur de ressources par le biais de InteractionSpec et l'authentification des composants par le biais de ConnectionSpec.

L'interface javax.cci.InteractionSpec transmet des informations sur le mode de traitement de la demande d'interaction avec l'adaptateur de ressources. Elle comporte également des informations sur l'accomplissement de l'interaction après la demande. Ces communications bidirectionnelles par le biais des interactions sont parfois appelées *conversations*.

La liaison EIS s'attend à ce que la charge qui sera l'argument de l'adaptateur de ressources contienne un objet de données enfant appelé **properties**. Cette objet de données de propriétés contient des paires nom/valeur, avec le nom des propriétés de la spécification d'interaction dans un format particulier. Les règles de formatage sont les suivantes :

- Les noms doivent commencer par le préfixe **IS**, suivi du nom de propriété. Par exemple, une spécification d'interaction avec une propriété JavaBeans appelée **InteractionId** doit spécifier le nom de la propriété sous la forme **ISInteractionId**.
- La paire nom/valeur représente le nom et la valeur du type simple de propriété de la spécification d'interaction.

Dans cet exemple, une interface spécifie que l'entrée d'une opération est un objet de données **Compte**. Cette interface appelle une application de liaison d'importation EIS dans le but d'envoyer et de recevoir une propriété InteractionSpec dynamique appelée **workingSet** avec la valeur **xyz**.

Le graphique métier ou les objets métier du serveur contiennent un objet métier **properties** sous-jacent qui permet l'envoi de données propres au protocole avec la charge. Cet objet métier**properties** est intégré, ce qui fait qu'il n'est pas nécessaire de le spécifier dans le schéma XML lors de la construction d'un objet métier. Il convient seulement de le créer et de l'utiliser. Si vous avez défini vos propres types de données sur la base du schéma XML, vous devez spécifier un élément **properties** qui contient les paires nom/valeur attendues.

```
BOFactory dataFactory = (BOFactory) \
serviceManager.locateService("com/ibm/websphere/bo/BOFactory");
 //Wrapper for doc-lit wrapped style interfaces,
 //skip to payload for non doc-lit
 DataObject docLitWrapper = dataFactory.createByElement /
 ("http://mytest/eis/Account", "AccountWrapper");
```

Créez la charge.

```
DataObject account = docLitWrapper.createDataObject(0);
  DataObject accountInfo = account.createDataObject("AccountInfo");
  //Perform your setting up of payload


  //Construct properties data for dynamic interaction

  DataObject properties = account.createDataObject("properties");
```

Pour le nom workingSet, définissez la valeur attendue (**xyz**).

```
properties.setString("ISworkingSet", "xyz");


  //Invoke the service with argument

  Service accountImport = (Service) \
  serviceManager.locateService("AccountOutbound");
  DataObject result = accountImport.invoke("createAccount", docLitWrapper);

  //Get returned property
  DataObject retProperties = result.getDataObject("properties");

  String workingset = retProperties.getString("ISworkingSet");
```

Vous pouvez utiliser des propriétés ConnectionSpec pour l'authentification des composants dynamiques. Les mêmes règles s'appliquent que ci-dessus, sauf que le préfixe du nom de propriété doit être **CS** au lieu de **IS**. Les propriétés ConnectionSpec ne sont pas bidirectionnelles. Le même objet de données **properties** peut contenir à la fois des propriétés IS et CS.

Pour utiliser les propriétés ConnectionSpec, définissez **resAuth** spécifié dans la liaison d'importation sur **Application**. Assurez-vous également que l'adaptateur de ressources prend en charge l'autorisation de composant. Pour plus de détails, reportez-vous au chapitre 8 du document J2EE Connector Architecture Specification.

Clients externes et liaisons EIS :

Le serveur peut envoyer des messages à, ou recevoir des messages de clients externes par le biais de liaisons EIS.

Un client externe, par exemple un portail Web ou un système EIS, doit envoyer un message à un module SCA dans le serveur ou doit être appelé par un composant à partir du serveur.

Le client appelle l'importation EIS comme avec toute autre application, à l'aide soit de l'interface DII (Dynamic Invocation Interface), soit de l'interface Java.

1. Le client externe crée une instance de l'interface ServiceManager et recherche l'importation EIS à l'aide de son nom de référence. Le résultat de la recherche est une implémentation de l'interface de service.
2. Le client crée un argument d'entrée, un objet de données générique, créé dynamiquement à l'aide du schéma de l'objet de données. Cette étape est effectuée à l'aide de l'implémentation de l'interface Service Data Object DataFactory.
3. Le client externe appelle l'EIS et obtient les résultats requis.

Le client a également la possibilité d'appeler l'importation EIS par le biais de l'interface Java.

1. Le client externe crée une instance de l'interface ServiceManager et recherche l'importation EIS à l'aide de son nom de référence. Le résultat de la recherche est une interface Java de l'importation EIS.
2. Le client crée un argument d'entrée et un objet de données typé.
3. Le client appelle l'EIS et obtient les résultats requis.

L'interface d'exportation EIS définit l'interface du composant SCA exporté qui est accessible aux applications EIS externes. Cette interface peut être considérée comme l'interface qu'une application externe (comme SAP ou PeopleSoft) va appeler par le biais de l'implémentation de l'environnement d'exécution de l'application d'exportation EIS.

L'exportation utilise EISExportBinding pour lier les services exportés à l'application EIS externe. Elle vous permet d'abonner une application contenue dans votre module SCA pour écouter les demandes de service EIS. La liaison d'exportation EIS spécifie le mappage entre la définition des événements entrants telle qu'elle est comprise par l'adaptateur de ressources (à l'aide des interfaces Java EE Connector Architecture) et l'appel des opérations SCA.

L'interface EISExportBinding exige que les services EIS externes soient basés sur des contrats entrants de Java EE Connector Architecture 1.5. L'interface EISExportBinding exige qu'un gestionnaire de données ou une liaison de données soit spécifié(e) au niveau de la liaison ou au niveau de la méthode.

Liaisons JMS

Un fournisseur JMS (Java Message Service) active la messagerie en fonction du modèle de programmation et de l'API JMS (Java Messaging Service). Il fournit des fabriques de connexions JMS pour créer des connexions pour des destinations JMS et pour envoyer et recevoir des messages.

Les liaisons JMS peut être utilisé lorsque vous interagissez avec la liaison du fournisseur du bus d'intégration de services (SIB) et si elles sont compatibles avec JMS et JCA 1.5.

Les liaisons d'importation et d'exportation JMS permettent à un module SCA (Service Component Architecture) d'appeler des systèmes JMS externes et d'en recevoir des messages.

Les liaisons d'importation et d'exportation JMS permettent une intégration aux applications JMS en utilisant le fournisseur JMS SIB basé sur JCA 1.5 qui fait partie de WebSphere Application Server. Les autres adaptateurs de ressources JMS basés sur JCA 1.5 ne sont pas pris en charge

Présentation des liaisons JMS :

Les liaisons JMS assurent la connectivité entre l'environnement SCA (Service Component Architecture) et les systèmes JMS.

Liaisons JMS

Les principaux composants des liaisons d'importation et d'exportation JMS sont les suivants :

- Adaptateur de ressources : assure une connectivité bidirectionnelle gérée entre un module SCA et des systèmes JMS externes
- Connexions : encapsulent une connexion virtuelle entre un client et une application fournisseur
- Destinations : utilisées par un client pour spécifier la cible des messages produits ou la source des messages utilisés
- Données d'authentification : permettent de sécuriser l'accès à la liaison

Principales fonctionnalités des liaisons JMS

En-têtes spéciaux

Des propriétés d'en-tête spéciales sont utilisées dans les importations et les exportations JMS pour indiquer à la cible comment traiter le message.

Par exemple, TargetFunctionName mappe la méthode native à la méthode d'opération.

Ressources Java EE

Plusieurs ressources Java EE sont créées lorsque les importations et les exportations JMS sont déployées dans un environnement Java EE.

`ConnectionFactory`
Utilisée par les clients pour créer une connexion au fournisseur JMS.

`ActivationSpec`
Utilisé par les importations pour la réception de la réponse à une demande et par les exportations pour la configuration endpoint des noeuds finaux de message qui représentent les écouteurs de messages dans leurs interactions avec le système de messagerie.

`Destinations`

- Destination d'envoi : Destination à laquelle est envoyé le message de demande ou sortant (importation) ou le message de réponse (exportation), si la valeur n'est pas remplacée par la zone d'en-tête JMSReplyTo du message entrant.
- Destination de réception : emplacement où doit être placé le message entrant ; dans les importations, il s'agit d'une réponse, dans les exportations, il s'agit d'une demande.
- Destination de rappel : Destination du système JMS SCA utilisée pour stocker les informations de corrélation. Vous ne devez pas procéder à des opérations de lecture ou d'écriture sur cette destination.

La tâche d'installation crée ConnectionFactory et trois destinations. En outre, elle crée ActivationSpec pour permettre à l'écouteur de messages d'exécution d'écouter les réponses sur la destination de réception. Les propriétés de ces ressources sont définies dans le fichier d'importation ou d'exportation.

Intégration JMS et adaptateurs de ressources :

Le service JMS (Java Message Service) assure une intégration par le biais d'un adaptateur de ressources JMS disponible basé sur JCA 1.5. La prise en charge complète de l'intégration JMS est assurée pour l'adaptateur de ressources JMS du bus d'intégration de services (SIB).

Utilisez un fournisseur JMS pour JCA 1.5 lorsque vous souhaitez une intégration avec un système JMS externe conforme au JCA 1.5. Les services externes conformes au JCA 1.5 peuvent recevoir et envoyer des messages à intégrer avec vos composants SCA à l'aide de l'adaptateur de ressources JMS SIB.

L'emploi d'adaptateurs de ressources JCA 1.5 propres à un fournisseur n'est pas pris en charge.

Liaisons d'importation et d'exportation JMS :

Vous pouvez faire interagir les modules SCA avec les services fournis par des applications JMS externes à l'aide des liaisons d'importation et d'exportation JMS.

Liaisons d'importation JMS

Les connexions avec le fournisseur JMS associé de destinations JMS sont créées à l'aide d'une fabrique de connexions JMS. Utilisez les objets d'administration de fabrique de connexions afin de gérer des fabriques de connexion JMS pour le fournisseur de messagerie par défaut.

L'interaction avec les systèmes JMS externes comprend l'utilisation des destinations pour l'envoi des requêtes et la réception des réponses.

Deux types de scénarios d'utilisation pour l'importation JMS sont pris en charge en fonction du type d'opération appelée :

- Unidirectionnel : l'importation JMS place un message sur la destination d'envoi configurée dans la liaison d'importation. Rien n'est défini dans la zone replyTo de l'en-tête JMS.
- Bidirectionnel (demande-réponse) : l'importation JMS place un message sur la destination d'envoi et conserve ensuite la réponse reçue depuis le composant SCA.

  La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut), ou à partir de l'ID de corrélation de message de demande. La liaison d'importation peut également être configurée pour utiliser une destination de réponse dynamique temporaire afin d'établir une corrélation entre les réponses et les demandes. Une destination temporaire est créée pour chaque demande et l'importation l'utilise pour recevoir la réponse.

  La destination receive est définie dans la propriété d'en-tête replyTo du message sortant. Un écouteur de messages est déployé pour écouter sur la destination de réception et, dès qu'une réponse est reçue, l'écouteur de messages transmet la réponse au composant.

Pour les scénarios d'utilisation unidirectionnel et bidirectionnel, les propriétés d'en-tête dynamique et statique peuvent être spécifiées. Les propriétés statiques peuvent être définies à partir de la liaison de méthode d'importation JMS. Certaines de ces propriétés revêtent des significations particulières pour l'environnement d'exécution JMS SCA.

Il est important de noter que JMS est une liaison asynchrone. Si un composant appelant appelle une importation JMS de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service JMS.

La figure 32, à la page 121 montre comment l'importation est liée au service externe.

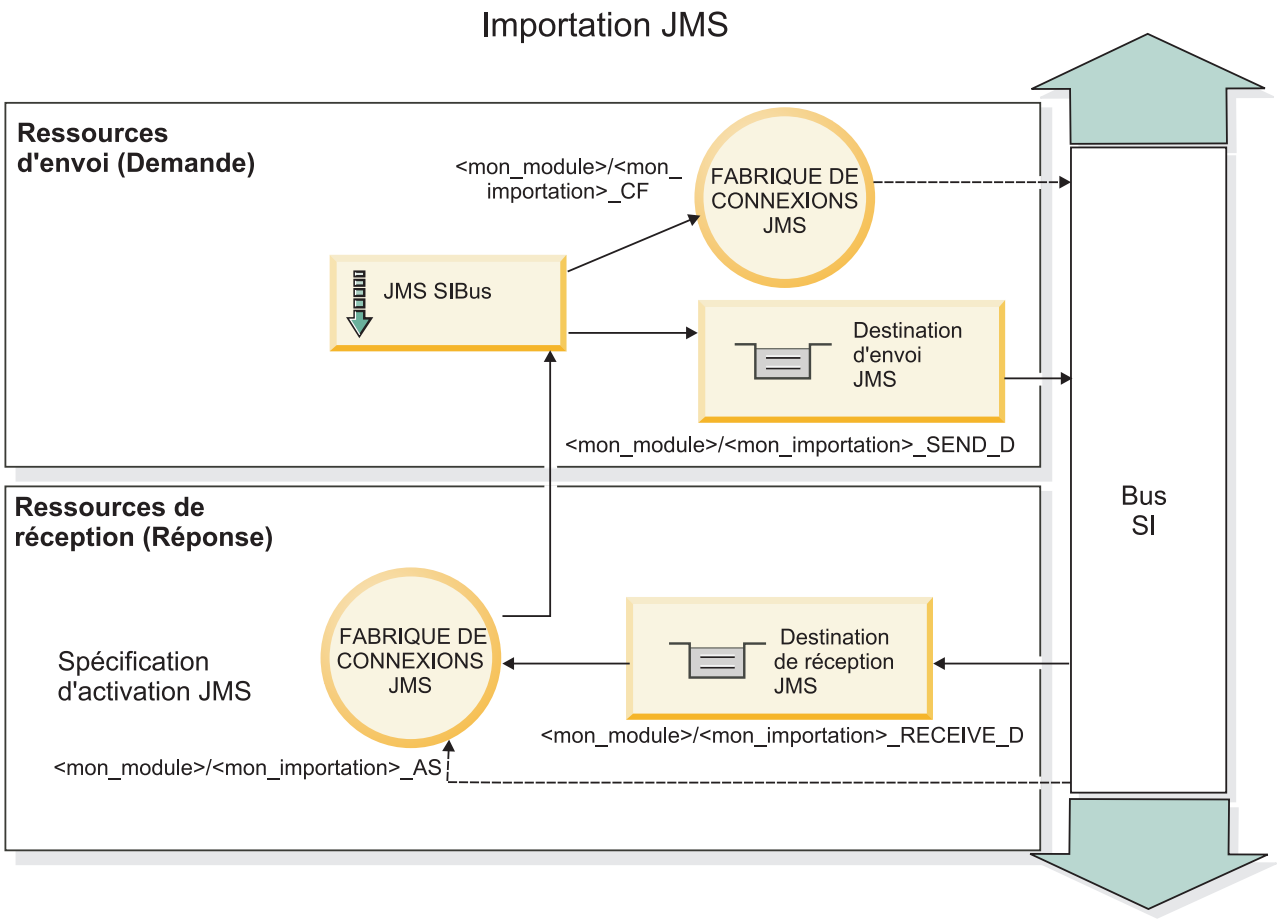


Figure 77. Ressources de liaisons d'importation JMS

Liaisons d'exportation JMS

Les liaisons d'exportation JMS offrent les moyens aux modules SCA de fournir des services aux applications JMS externes.

La connexion qui fait partie d'une exportation JMS est une spécification d'activation.

Une exportation JMS comporte des destinations d'envoi et de réception.

- La destination receive est le lieu de réception du message entrant destiné au composant cible.
- La destination send est celle à laquelle la réponse sera envoyée, sauf si le message entrant l'a remplacé en utilisant la propriété d'en-tête replyTo.

Un écouteur de messages est déployé pour écouter les demandes parvenant à la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse. La destination spécifiée dans la zone replyTo du message entrant remplace la destination spécifiée dans send.

La figure 33, à la page 122 illustre la manière dont le demandeur externe est lié à l'exportation.

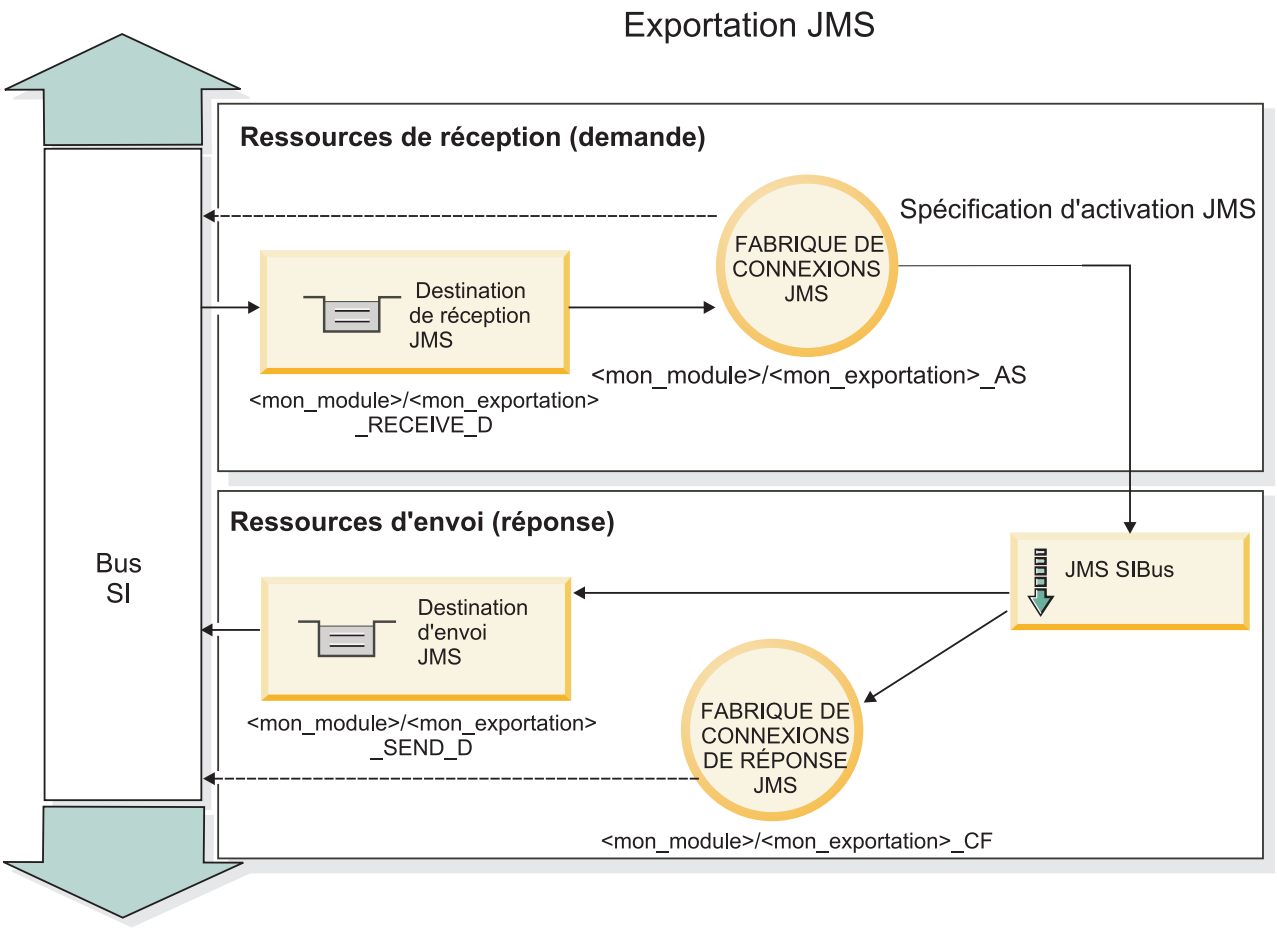


Figure 78. Ressources de liaisons d'exportation JMS

En-têtes JMS :

Un message JMS contient deux types d'en-têtes : l'en-tête de système JMS et plusieurs propriétés JMS. Il est possible d'accéder aux deux types d'en-tête depuis un module de médiation dans l'objet SMO (Service Message Object) ou en utilisant l'API ContextService.

En-tête système JMS

L'en-tête système JMS est représenté dans l'objet SMO par l'élément JMSHeader qui contient toutes les zones généralement présentes dans un en-tête JMS. Bien qu'elles puissent être modifiées dans la médiation (ou ContextService), certaines zones d'en-tête système JMS définies dans l'objet SMO ne seront pas propagées dans le message JMS sortant puisqu'elles sont remplacées par des valeurs statiques ou système.

Les zones clés de l'en-tête système JMS qui peuvent être mises à jour dans une médiation (ou ContextService) sont :

- **JMSType** et **JMSCorrelationID** : valeurs des propriétés de l'en-tête du message prédéfinies spécifiques
- **JMSDeliveryMode** : valeurs du mode de livraison (persistant (par défaut) ou non persistant)
- **JMSPriority** : valeur de la priorité (0 à 9 ; la valeur par défaut est JMS_Default_Priority)

Propriétés JMS

Les propriétés JMS sont représentées dans l'objet SMO en tant qu'entrées dans la liste Propriétés. Les propriétés peuvent être ajoutées, mises à jour ou supprimées dans une médiation ou en utilisant l'API ContextService.

Elles peuvent également être définies de manière statique dans la liaison JMS. Les propriétés définies de manière statique remplacent les paramètres (portant le même nom) qui sont définis de manière dynamique.

Les propriétés utilisateur propagées à partir d'autres liaisons (par exemple, une liaison HTTP) correspondront à une sortie dans la liaison JMS comme les propriétés JMS.

Paramètres de propagation d'en-tête

La propagation des propriétés et des en-têtes système JMS depuis le message JMS entrant vers les composants en aval ou depuis les composants en amont vers le message JMS sortant peut être contrôlée par l'indicateur Propagate Protocol Header.

Lorsque Propagate Protocol Header est défini, les informations d'en-tête peuvent circuler vers le message ou vers le composant cible, comme indiqué dans la liste suivante :

- Requête d'exportation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.
- Réponse d'exportation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS.
- Requête d'importation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS.
- Réponse d'importation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.

Schéma de corrélation des destinations de réponse dynamique temporaires JMS :

Le schéma de corrélation des destinations de réponse dynamique temporaires crée une rubrique ou une file d'attente dynamique unique pour chaque demande envoyée.

La destination de réponse statique spécifiée dans l'importation permet de déterminer la nature de la rubrique ou de la file d'attente de destination temporaire. Elle est définie dans la zone **ReplyTo** de la demande et l'importation JMS écoute les réponses sur cette destination. Lors de la réception d'une réponse, cette dernière est replacée dans la file d'attente de la destination des réponses statiques pour un traitement asynchrone. La zone **CorrelationID** de la réponse n'est pas utilisée et n'a pas besoin d'être définie.

Incidents transactionnels

Si une destination dynamique temporaire est utilisée, la réponse doit être consommée dans la même unité d'exécution que la réponse envoyée. La demande doit être envoyée en dehors de la transaction globale et doit être validée avant d'être reçue par le service dorsal et avant qu'une réponse ne soit renvoyée.

Persistance

Les files d'attente dynamiques temporaires sont des entités dont la durée de vie est courte et ne garantissent pas le même niveau de persistance que celui associé à une rubrique ou une file d'attente statique. Une rubrique ou une file d'attente dynamique temporaire ne survit pas à un redémarrage du serveur, tout comme les messages. Une fois que le message a été replacé en file d'attente dans la destination de réponse statique, elle conserve la persistance définie dans le message.

Délai d'attente

L'importation attend de recevoir la réponse sur la destination de réponse dynamique temporaire pendant un certain temps. Ce délai est extrait du qualifiant de délai d'expiration des réponses SCA, s'il est défini. Dans le cas contraire, le délai par défaut est de 60 secondes. En cas de dépassement de ce délai, l'importation génère une erreur `ServiceTimeoutRuntimeException`.

Clients externes :

Le serveur peut envoyer des messages à, ou recevoir des messages de clients externes par le biais de liaisons JMS.

Un client externe (comme un portail Web ou un système d'information d'entreprise) peut envoyer un message à un module SCA dans le serveur ou peut être appelé par un composant à partir du serveur.

Les composants d'exportation JMS déploient des écouteurs des messages pour écouter les demandes entrantes sur la destination receive spécifié dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si l'application appelée fournit une réponse. Ainsi, un client externe est en mesure d'appeler des applications avec la liaison d'exportation.

Les importations JMS interagissent avec les clients externes en envoyant des messages aux files d'attente JMS et en recevant des messages de ces dernières.

Utilisation de clients externes :

Un client externe (c'est-à-dire extérieur au serveur) peut avoir besoin d'interagir avec une application installée sur le serveur.

Le scénario présenté ici est très simple : un client externe souhaite interagir avec une application sur le serveur. La figure représente un scénario simple typique.

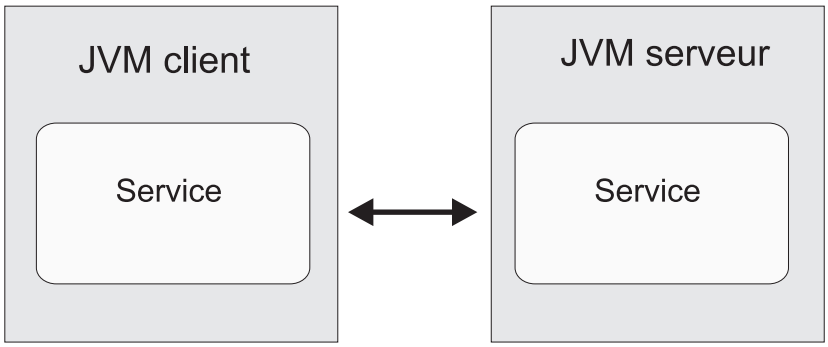


Figure 79. Scénario simple : interaction entre un client externe et une application du serveur

L'application SCA inclut une exportation avec une liaison JMS ; c'est ce qui la rend disponible aux clients externes.

Si un client externe se trouve sur une machine Java virtuelle (JVM) distincte de votre serveur, vous devez effectuer plusieurs opérations afin d'établir la connexion et permettre l'interaction avec une exportation JMS. Le client obtient un contexte initial (InitialContext) avec les valeurs appropriées, puis il compulse les ressources via JNDI. Le client utilise alors le client de spécification JMS 1.1 pour accéder aux destinations et envoyer/recevoir des messages sur les destinations.

Les noms JNDI par défaut des ressources créées automatiquement par l'environnement d'exécution sont répertoriés dans la rubrique configuration de cette section. Toutefois, si vous avez pré-créé les ressources, utilisez ces noms JNDI.

1. Configurer les destinations JMS et la fabrique de connexions pour envoyer le message.
2. Vérifier que le contexte JNDI, le port de l'adaptateur de ressources SIB et le port d'amorçage de la messagerie sont corrects.

   Le serveur utilise certains ports par défaut, mais si plusieurs serveurs sont installés sur ce système, des ports de remplacement sont créés lors de l'installation pour éviter les conflits avec d'autres instances de serveur. Vous pouvez utiliser la console d'administration pour déterminer quels ports votre serveur employer. Accédez à **Serveurs** > **Serveurs d'applications** > *nom_serveur* > **Configuration** et cliquez sur **Ports** sous **Communication**. Vous pouvez maintenant modifier le port utilisé.
3. Le client obtient un contexte initial avec les valeurs appropriées, puis il compulse les ressources via JNDI.
4. A l'aide des spécifications JMS 1.1, le client accède aux destinations et aux messages envoyés et reçus sur les destinations.

Identification et résolution des incidents liés aux liaisons JMS :

Vous pouvez diagnostiquer et résoudre des incidents survenant sur les liaisons JMS.

Exceptions liées à l'implémentation

En réponse à diverses conditions d'erreur, l'implémentation de l'importation et de l'exportation JMS peut renvoyer deux types d'exception :

- Service : cette exception est renvoyée si l'erreur définie sur l'interface métier de service (type de port WSDL) s'est produite.
- Service Runtime : générée dans tous les autres cas. Dans la plupart des cas, l'exception cause contient l'exception d'origine (JMSException).

  Par exemple, une importation n'attend qu'un seul message de réponse pour chaque message de demande. Si plusieurs réponses sont fournies, ou si une réponse tardive (une pour laquelle la réponse

SCA a expiré) est fournie, une exception Service Runtime est générée. La transaction est annulée et le message de réponse est retiré de la file d'attente ou traité par le gestionnaire d'événements ayant échoué.

Conditions d'erreur principales

Les principales conditions d'erreur liées aux liaisons JMS sont déterminées par la sémantique transactionnelle, la configuration du fournisseur JMS ou une référence au fonctionnement existant dans d'autres composants. Les causes premières d'incident peuvent être :

- Erreur de connexion au fournisseur ou à la destination JMS.

  Une erreur de connexion au fournisseur JMS pour recevoir des messages entraîne l'échec de démarrage de l'écouteur des messages. Cette situation sera enregistrée dans la journal WebSphere Application Server. Des messages persistants resteront sur la destination jusqu'à ce qu'ils soient récupérés (ou qu'ils parviennent à échéance).

  Une erreur de connexion au fournisseur JMS destinée à envoyer des messages entraîne l'annulation de la transaction qui contrôle l'envoi.
- Erreur d'analyse d'un message entrant ou de construction d'un message sortant.

  Une erreur de liaison des données ou du gestionnaire de données entraîne l'annulation de la transaction qui contrôle le travail.
- Erreur d'envoi du message sortant.

  Un échec d'envoi d'un message entraîne l'annulation de la transaction en question.
- Messages de réponse multiples ou inattendus.

  L'importation n'attend qu'un seul message de réponse pour chaque message de demande. Egalement la période valide durant laquelle une réponse peut être reçue par le qualifiant d'expiration de réponse SCA sur la demande. Lorsqu'une réponse est reçue ou le délai d'expiration expire, l'enregistrement de corrélation est supprimé. Si des messages de réponse parviennent de manière inattendue ou tardive, une exception Service Runtime est générée.
- Exception d'exécution lié au délai d'expiration du service, générée par une réponse tardive lors de l'utilisation du schéma de corrélation des destinations de réponse dynamique temporaires.

  L'importation JMS arrive à expiration après un délai déterminé par le qualifiant d'expiration des réponses SCA ; si ce délai n'est pas défini, il est de 60 secondes par défaut.

Messages SCA basés sur JM qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si les messages SCA émis par le biais d'une interaction JMS échouent, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Or, si ces messages n'apparaissent pas, vérifiez que la destination SIB sous-jacente de la destination JMS possède une valeur du nombre maximal de livraisons ayant échoué supérieure à **1**. Définir cette valeur sur **2** ou plus permet une interaction avec le gestionnaire des événements ayant échoué au cours des appels SCA pour les liaisons JMS.

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS, par défaut, est stocké dans le gestionnaire des événements ayant échoué pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Liaisons JMS génériques

La liaison JMS générique assure la connectivité avec les fournisseurs tiers compatibles avec JMS 1.1. Le fonctionnement des liaisons JMS génériques est identique à celui des liaisons JMS.

Le service fourni par le biais d'une liaison JMS permet à un module SCA (Service Component Architecture) d'effectuer des appels ou de recevoir des messages à partir de systèmes externes. Le système peut être un système JMS externe.

La liaison JMS générique assure l'intégration avec les fournisseurs JMS conformes non JCA 1.5 qui prennent en charge JJMS 1.1 et implémentent la fonction de serveur d'applications JMS disponible en option. La liaison JMS générique prend en charge les fournisseurs JMS (notamment Oracle AQ, TIBCO, SonicMQ, WebMethods et BEA WebLogic) qui ne prennent pas en charge JCA 1.5 mais qui prennent en charge la fonction de serveur d'applications de la spécification JMS 1.1. Le fournisseur JMS imbriqué dans WebSphere (SIBJMS), qui est un fournisseur JMS JCA 1.5, n'est pas pris en charge par cette liaison ; si vous utilisez ce fournisseur, utilisez les «Liaisons JMS», à la page 118.

Cette liaison générique peut par exemple être utilisée lors d'une intégration avec un système JMS conforme non JCA 1.5 dans un environnement. Les applications externes cibles peuvent alors recevoir et envoyer des messages à intégrer avec un composant SCA.

Présentation des liaisons JMS génériques :

Les liaisons JMS génériques sont des liaisons JMS non JCA qui assurent la connectivité entre l'environnement SCA (Service Component Architecture) et les systèmes JMS compatibles avec JMS 1.1 et qui implémentent la fonction de serveur d'applications JMS en option.

Liaisons JMS génériques

Les aspects majeurs des liaisons d'importation et d'exportation JMS génériques sont les suivants :

- Port de l'écouteur : permet aux fournisseurs JMS non JCA de recevoir des messages et de les distribuer à un bean géré par message
- Connexions : encapsulent une connexion virtuelle entre un client et une application fournisseur
- Destinations : utilisées par un client pour spécifier la cible des messages produits ou la source des messages utilisés
- Données d'authentification : permettent de sécuriser l'accès à la liaison

Liaisons d'importation JMS génériques

Les liaisons d'importation JMS génériques permettent aux composants au sein de votre module SCA de communiquer avec les services fournis par les fournisseurs JMS conformes non JCA 1.5.

La connexion qui fait partie d'une importation JMS est une fabrique de connexions. Une fabrique de connexion, l'objet utilisé par un client pour créer une connexion à un fournisseur, encapsule un ensemble de paramètres de configuration de la connexion définie par un administrateur. Chaque fabrique de connexions est une instance de l'interface ConnectionFactory, QueueConnectionFactory ou TopicConnectionFactory.

L'interaction avec les systèmes JMS externes comprend l'utilisation des destinations pour l'envoi des requêtes et la réception des réponses.

Deux types de scénarios d'utilisation pour la liaison d'import ation JMS générique sont pris en charge, en fonction du type d'opération appelée :

- Unidirectionnel : l'importation JMS générique place un message sur la destination d'envoi configurée dans la liaison d'importation. Rien n'est envoyé à la zone replyTo de l'en-tête JMS.
- Bidirectionnel (demande-réponse) : l'importation JMS générique place un message sur la destination d'envoi et conserve ensuite la réponse reçue depuis le composant SCA.

  La destination receive est définie dans la propriété d'en-tête replyTo du message sortant. Un bean géré par message (MDB) est déployé pour écouter sur la destination de réception et, dès qu'une réponse est reçue, le MDB transmet la réponse au composant.

  La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut) ou à partir de l'ID de corrélation de message de demande.

Pour les scénarios d'utilisation unidirectionnel et bidirectionnel, les propriétés d'en-tête dynamique et statique peuvent être spécifiées. Les propriétés statiques peuvent être définies à partir de la liaison de méthode d'importation JMS générique. Certaines de ces propriétés revêtent des significations particulières pour l'environnement d'exécution JMS SCA.

Il est important de noter que la liaison JMS générique est une liaison asynchrone. Si un composant appelant appelle une importation JMS générique de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service JMS.

La figure 35, à la page 129 illustre la manière dont l'importation est liée au service externe.

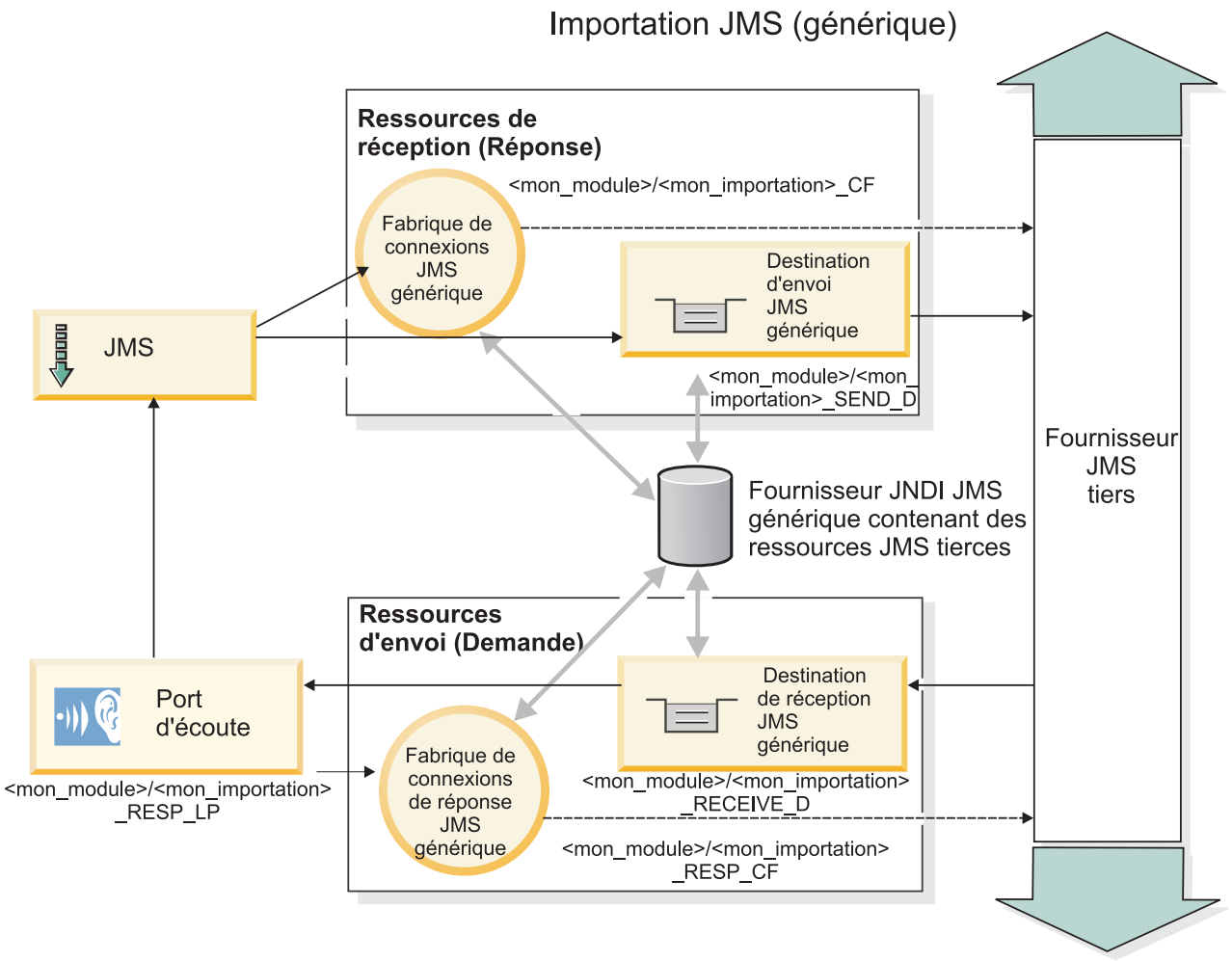


Figure 80. Ressources de liaisons d'importation JMS génériques

Liaisons d'exportation JMS génériques

Les liaisons d'exportation JMS génériques offrent les moyens aux modules SCA de fournir des services aux applications JMS externes.

La connexion qui fait partie d'une exportation JMS est composée d'une entité ConnectionFactory et d'une entité ListenerPort.

Une exportation JMS générique comporte des destinations d'envoi et de réception.

- La destination receive est le lieu de réception du message entrant destiné au composant cible.
- La destination send est celle à laquelle la réponse sera envoyée, sauf si le message entrant l'a remplacé en utilisant la propriété d'en-tête replyTo.

Un bean MDB est déployé pour écouter les demandes parvenant à la destination receive spécifiée dans la liaison d'exportation.

- La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse.
- La destination spécifiée dans la zone replyTo du message entrant remplace la destination spécifiée dans la zone send.
- Pour les scénarios de demande/réponse, la liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que la réponse

copie l'ID de message de demande dans la zone ID de corrélation du message de réponse (option par défaut) ou la réponse peut copier l'ID de corrélation de la demande dans la zone ID de corrélation du message de réponse.

La figure 36, à la page 130 illustre la manière dont le demandeur externe est lié à l'exportation.

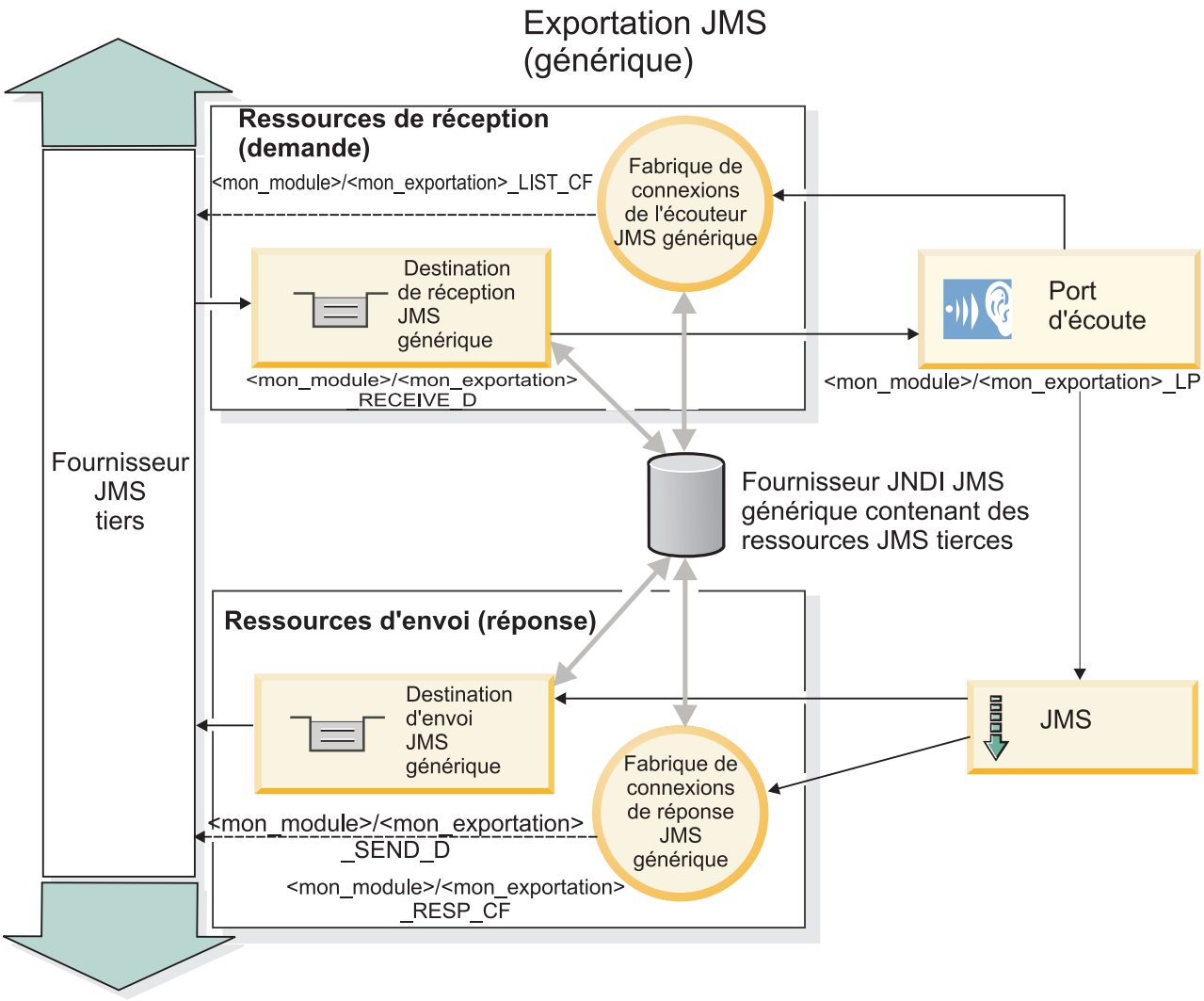


Figure 81. Ressources de liaisons d'exportation JMS génériques

Principales fonctionnalités des liaisons JMS Generic :

Les fonctions de la liaison Generic d'importation et d'exportation JMS correspondent aux fonctions des liaisons d'importation JMS et MQ JMS intégrées à WebSphere. Les fonctions essentielles incluent les définitions d'en-tête et l'accès aux ressources Java EE existantes. Cependant, étant donné son caractère générique, cette liaison ne comporte pas d'options de connectivité spécifiques au fournisseur JMS et elle est limitée à la génération de ressources lors du déploiement et de l'installation.

Importations génériques

Comme l'application d'importation MQ JMS, l'implémentation JMS Generic est asynchrone et prend en charge trois types d'appel : unidirectionnel, bidirectionnel (également appelé demande-réponse) et rappel.

Si l'importation JMS est déployée, un bean géré par message (MDB) fourni par l'environnement d'exécution est déployé. Le bean MDB est à l'écoute des réponses au message de demande. Le bean MDB est associé à (écoute) la destination envoyée avec la demande dans la zone d'en-tête replyTo du message JMS.

Exportations génériques

Les liaisons d'exportation JMS génériques diffèrent des liaisons d'exportation EIS au niveau du traitement du renvoi du résultat. Une exportation JMS Generic envoie explicitement la réponse à la destination replyTo spécifiée dans le message entrant. Si aucune destination n'est indiquée, la destination d'envoi est utilisée.

Lorsque l'exportation JMS Generic est déployée, un bean MDB (autre que le bean MDB utilisé pour les importations JMS Generic) est déployé. Il écoute les requêtes entrantes sur la destination de réception, puis distribue les requêtes que doit traiter l'environnement d'exécution SCA.

En-têtes spéciaux

Des propriétés d'en-tête spéciales sont utilisées dans les importations et les exportations JMS Generic pour indiquer à la cible comment traiter le message.

Par exemple, la propriété TargetFunctionName permet au sélecteur de fonction par défaut d'identifier le nom de l'opération sur l'interface d'exportation appelée.

Remarque : Il est possible de configurer la liaison d'importation pour que l'en-tête TargetFunctionName porte le nom de l'opération, dans chaque cas.

Ressources Java EE

Plusieurs ressources Java EE sont créées lors du déploiement d'une liaison JMS dans un environnement Java EE.

- Port d'écoute sur la destination de réception (réponse) (bidirectionnel uniquement) pour les importations, et sur la destination de réception (demande) pour les exportations
- Fabrique de connexions JMS générique pour outboundConnection (importation) et inboundConnection (exportation)
- Destination JMS générique pour les destinations d'envoi (importation) et de réception (exportation) (bidirectionnel uniquement)
- Fabrique de connexions JMS générique pour responseConnection (bidirectionnel uniquement et facultatif), sinon : outboundConnection est utilisé pour les importations et inboundConnection est utilisé pour les exportations)
- Destination JMS générique pour la destination de réception (importation) et d'envoi (exportation) (bidirectionnel uniquement)
- Destination JMS de rappel de fournisseur de messagerie par défaut utilisée pour l'accès à la destination de file d'attente de rappel SIB (bidirectionnel uniquement)
- Fabrique de connexions JMS de rappel de fournisseur de messagerie par défaut utilisée pour l'accès à la destination JMS de rappel (bidirectionnel uniquement)
- Destination de file d'attente de rappel SIB utilisée pour stocker les informations sur le message de demande à utiliser lors du traitement de la réponse (bidirectionnel uniquement)

La tâche d'installation crée ConnectionFactory, les trois destinations et ActivationSpec à partir des informations des fichiers d'importation et d'exportation.

En-têtes JMS génériques :

Les en-têtes JMS génériques sont des objets SDO (Service Data Objects) qui contiennent toutes les propriétés des propriétés de message JMS génériques. Ces propriétés peuvent provenir du message entrant ou il peut s'agir des propriétés qui seront appliquées au message sortant.

Un message JMS contient deux types d'en-têtes : l'en-tête de système JMS et plusieurs propriétés JMS. Il est possible d'accéder aux deux types d'en-tête depuis un module de médiation dans l'objet SMO (Service Message Object) ou en utilisant l'API ContextService.

Les propriétés suivantes sont définies statiquement sur methodBinding :

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

La liaison JMS générique JMS prend également en charge la modification dynamique des propriétés et des en-têtes JMS de la même manière que les liaisons JMS et JMS MQ.

Certains fournisseurs JMS génériques instaurent des restrictions sur quelles propriétés pouvant être définies par l'application et avec quelles combinaisons. Pour plus d'informations, reportez-vous à la documentation de votre produit tiers. Toutefois, une propriété complémentaire a été ajoutée à methodBinding, ignoreInvalidOutboundJMSProperties, qui permet de propager n'importe quelle exception.

Les propriétés de message et d'en-tête JMS génériques sont uniquement utilisées lorsque le commutateur de liaisons SCDL de l'architecture de composants de service de base est sous tension. Lorsque le commutateur est activé, les informations contextuelles sont propagées. Par défaut, ce commutateur est sous tension. Pour éviter la diffusion des informations contextuelles, remplacez la valeur par **false**.

Lorsque la diffusion du contexte est activée, les informations d'en-tête peuvent être transmises au message ou au composant cible. Pour activer et désactiver la diffusion du contexte, spécifiez la valeur **true** ou **false** pour l'attribut contextPropagationEnabled des liaisons d'importation et d'exportation. Exemple :

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

La valeur par défaut est **true**.

Résolution d'incidents liés aux liaisons JMS génériques :

Vous pouvez diagnostiquer et résoudre des incidents survenant sur les liaisons JMS génériques.

Exceptions liées à l'implémentation

En réponse à diverses conditions d'erreur, l'implémentation de l'importation et de l'exportation JMS générique peut renvoyer deux types d'exception :

- Service : cette exception est renvoyée si l'erreur définie sur l'interface métier de service (type de port WSDL) s'est produite.
- Service Runtime : générée dans tous les autres cas. Dans la plupart des cas, l'exception cause contient l'exception d'origine (JMSException).

Résolution d'une expiration de message JMS générique

Expiration d'un message de demande du fournisseur JMS.

L'*expiration de la demande* désigne l'expiration d'un message de demande du fournisseur JMS à la fin du délai JMSExpiration indiqué dans le message de demande. Comme pour d'autres liaisons JMS, la liaison JMS générique traite l'expiration de la demande en attribuant à l'expiration du message de rappel placé lors de l'importation la même valeur que celle de la demande sortante. La notification de l'expiration du message de rappel indique que le message de demande a expiré et le client doit être informé au moyen d'une exception.

Cependant, si la destination de rappel est transférée sur le fournisseur tiers, ce type d'expiration de demande n'est pas pris en charge.

L'*expiration de la réponse* désigne l'expiration d'un message de réponse du fournisseur JMS à la fin du délai JMSExpiration indiqué dans le message de réponse.

L'expiration de la réponse n'est pas prise en charge pour la liaison JMS générique, car le fonctionnement d'une expiration sur un fournisseur JMS tiers n'est pas définie. Toutefois, vous pouvez déterminer si la réponse a expiré lors de sa réception.

Pour les messages de demande sortants, la valeur de JMSExpiration est calculée à partir du temps d'attente et des valeurs requestExpiration indiquées dans asyncHeader, si elles sont définies.

Résolution d'erreurs liées aux fabriques de connexion JMS génériques

Lorsque vous définissez certains types de fabriques de connexion au niveau du fournisseur JMS générique, vous pouvez recevoir un message d'erreur lors d'une tentative de lancement d'une application. Vous pouvez modifier la fabrique de connexions externe pour empêcher ce problème de se produire.

Lorsque vous lancez une application, le message suivant peut s'afficher :

```
Le type JMSConnectionFactory du port d'écoute MDB ne correspond pas
au type JMSDestination
```

Cet incident peut survenir lorsque vous définissez des fabriques de connexions externes. En particulier, l'exception peut être générée lorsque vous créez une fabrique de connexions de sujet JMS 1.0.2 au lieu d'une fabrique de connexions JMS 1.1 (unifiée) (c'est-à-dire, une fabrique qui prenne en charge les communications de type point à point et publication/abonnement).

Pour résoudre ce problème, procédez comme suit :

1. Accédez au fournisseur JMS générique que vous utilisez.
2. Remplacez la fabrique de connexions de sujet JMS 1.0.2 que vous avez définie par une fabrique de connexions JMS 1.1 (unifiée).

Lorsque vous lancez l'application avec la nouvelle fabrique de connexions JMS 1.1, le message d'erreur ne devrait plus s'afficher.

Messages SCA génériques basés sur JMS qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si les messages SCA émis par le biais d'une interaction JMS générique échouent, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Si ces messages n'apparaissent pas, vérifiez que la valeur de la propriété du nombre maximal de nouvelles tentatives sur le port d'écoute sous-jacent est supérieure ou égale à 1. Définir cette valeur sur 1 ou plus permet une interaction avec le gestionnaire d'événements ayant échoué au cours des appels SCA pour les liaisons JMS génériques.

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS est stocké dans le gestionnaire des événements ayant échoué par défaut pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Le gestionnaire de données étant appelé par la liaison de données, une exception de gestionnaire de données est générée dans une exception de liaison de données. Par conséquent, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Liaisons JMS WebSphere MQ

La liaison JMS WebSphere MQ assure l'intégration avec les applications externes qui utilisent un fournisseur basé sur JMS WebSphere MQ.

Utilisez les liaisons d'importation et d'exportationJMS WebSphere MQ lorsque vous souhaitez une intégration directe avec des systèmes JMS ou JMS MQ à partir de votre environnement de serveur. Ainsi, il n'est plus nécessaire d'utiliser les fonctions de lien MQ ou client du bus d'intégration de services.

Lorsqu'un composant interagit avec un service basé sur JMS WebSphere MQ par le biais d'une importation, la liaison d'importation JMS WebSphere MQ utilise une destination vers laquelle les données seront envoyées et une destination à laquelle la réponse peut être reçue. La conversion des données vers ou depuis un message JMS est accomplie via le composant Edge Component du gestionnaire de données ou de la liaison de données.

Lorsqu'un module SCA fournit un service aux clients JMS WebSphere MQ, la liaison d'exportation JMS WebSphere MQ utilise une destination à laquelle la requête peut être reçue et la réponse envoyée. La conversion des données vers et depuis un message JMS s'effectue par le biais du gestionnaire de données ou de la liaison de données JMS.

Le sélecteur de fonction sert à effectuer un mappage avec l'opération sur le composant cible à appeler.

Présentation des liaisons JMS WebSphere MQ :

La liaison JMS WebSphere MQ assure l'intégration aux applications externes qui utilisent le fournisseur JMS WebSphere MQ.

Tâches d'administration WebSphere MQ

Avant d'exécuter une application contenant ce type de liaison, l'administrateur système WebSphere MQ est censé créer le gestionnaire de files d'attente WebSphere MQ sous jacent qui sera utilisé par les liaisons JMS WebSphere MQ.

Liaisons d'importation JMS WebSphere MQ

L'importation JMS WebSphere MQ permet aux composants au sein de votre module SCA de communiquer avec des services offerts par les fournisseurs basés sur JMS WebSphere MQ. Vous devez utiliser une version prise en charge de WebSphere MQ. Pour plus de détails sur la configuration logicielle et matérielle requise, reportez-vous aux pages de support IBM..

Deux types de scénarios d'utilisation pour les liaisons d'importation JMS WebSphere MQ JMS sont pris en charge, en fonction du type d'opération appelée :

- Unidirectionnel : l'importation JMS WebSphere MQ place un message sur la destination d'envoi configurée dans la liaison d'importation. Rien n'est envoyé à la zone replyTo de l'en-tête JMS.
- Bidirectionnel (demande-réponse) : l'importation JMS WebSphere MQ place un message sur la destination d'envoi.

  La destination receive est définie dans la zone d'en-tête replyTo. Un bean géré par message (MDB) est déployé pour écouter sur la destination de réception et, dès qu'une réponse est reçue, le MDB transmet la réponse au composant.

  La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse** dans Integration Designer) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut) ou à partir de l'ID de corrélation de message de demande.

Pour les scénarios d'utilisation unidirectionnels et bidirectionnels, les propriétés d'en-tête dynamiques et statiques peuvent être spécifiées.Les propriétés statiques peuvent être définies à partir de la liaison de méthode d'importation JMS. Certaines de ces propriétés revêtent des significations particulières pour l'environnement d'exécution JMS SCA.

Il est important de noter que JMS WebSphere MQ est une liaison asynchrone. Si un composant appelant appelle une importation JMS WebSphere MQ de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service.

La figure 37, à la page 136 montre comment l'importation est liée au service externe.

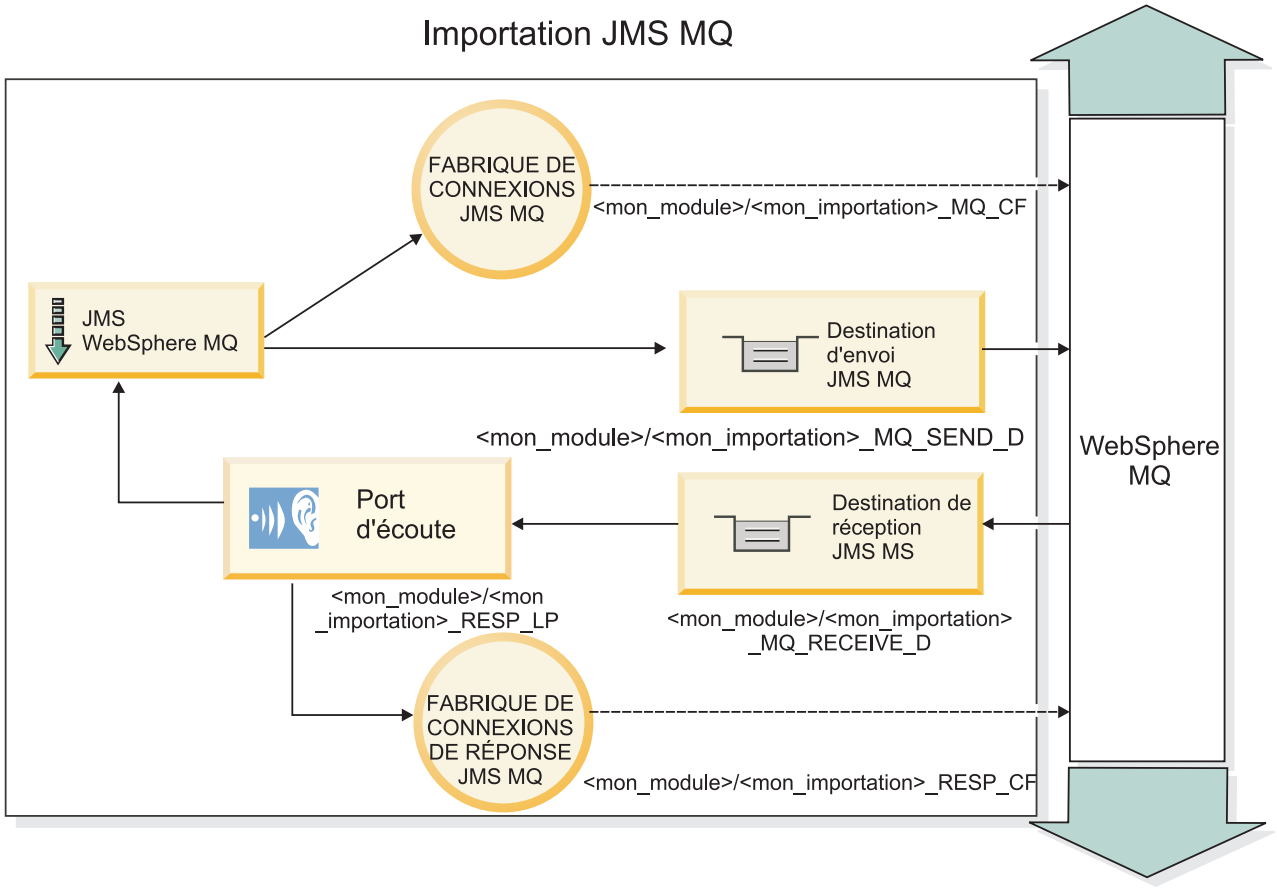


Figure 82. Ressources de liaison d'importation JMS WebSphere MQ

Liaisons d'exportation JMS Liaisons JMS WebSphere MQ

La liaison d'exportation JMS WebSphere MQ offre les moyens aux modules SCA de fournir des services aux applications JMS externes sur le fournisseur JMS basé sur WebSphere MQ.

Un bean MDB est déployé pour écouter les demandes parvenant à la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse. La destination spécifiée dans la zone replyTo du message de réponse remplace la destination spécifiée dans la zone send.

La figure 38, à la page 137 illustre la manière dont le demandeur externe est lié à l'exportation.

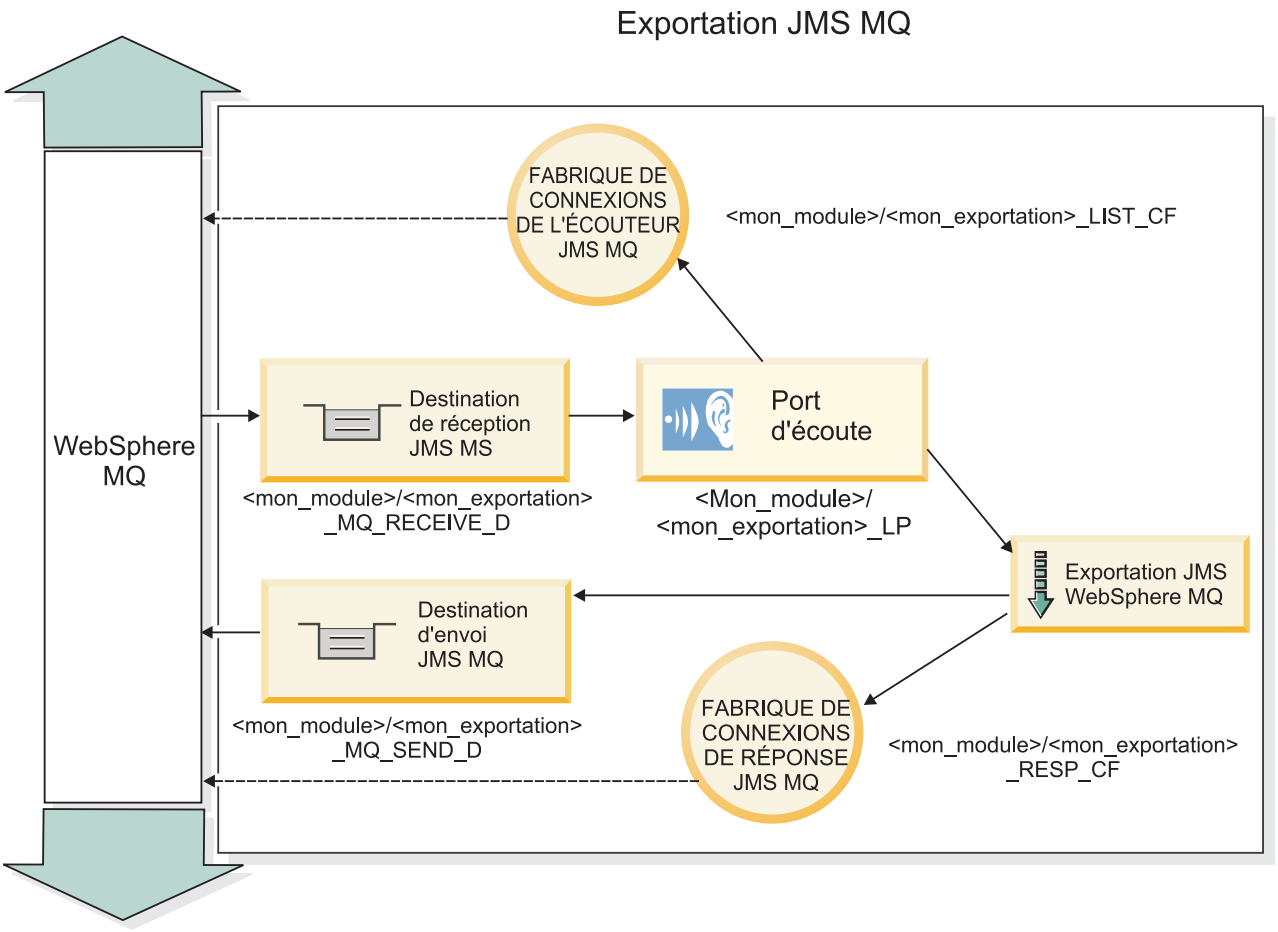


Figure 83. Ressources de liaison d'exportation JMS WebSphere MQ

Remarque : La figure 37, à la page 136 et la figure 38, à la page 137 illustrent comment une application d'une version précédente de IBM Business Process Manager est associée à un service externe. Pour les applications développées pour IBM Business Process Manager, version 7.0, la spécification d'activation est utilisée à la place du port de l'écouteur et de la fabrique de connexions.

Principales fonctionnalités des liaisons JMS WebSphere MQ :

Les fonctionnalités essentielles des liaisons JMS WebSphere MQ incluent les en-têtes, les artefacts Java EE et les ressources Java EE créées.

En-têtes

Un en-tête de message JMS contient plusieurs zones prédéfinies qui contiennent des valeurs utilisées par les clients et les fournisseurs pour identifier et acheminer les messages. Vous pouvez utiliser les propriétés de liaison pour configurer ces en-têtes avec des valeurs fixes, ou vous pouvez spécifier dynamiquement les en-têtes lors de l'exécution.

`JMSCorrelationID`
Liaison avec un message associé. En général, cette zone est définie sur la chaîne de l'identificateur du message auquel une réponse est envoyée.

`TargetFunctionName`
Cet en-tête est utilisé par l'un des sélecteurs de fonction fournis pour identifier l'opération appelée. La définition de la propriété d'en-tête JMS TargetFunctionName dans des messages envoyés à une exportation JMS permet l'utilisation de ce sélecteur de fonction. La propriété peut être définie

directement dans les applications client JMS ou lors de la connexion d'une importation avec une liaison JMS à une exportation de ce type. Dans ce cas, vous devez configurer la liaison d'importation JMS pour que l'en-tête TargetFunctionName pour chaque opération de l'interface soit défini sur le nom de l'opération.

Schémas de corrélation

Les liaisons JMS WebSphere MQ fournissent un grand nombre de schémas de corrélation qui permettent de déterminer la manière de corréler les messages de demande avec les messages de réponse.

RequestMsgIDToCorrelID
JMSMessageID est copié dans la zone JMSCorrelationID. Il s'agit du paramètre par défaut.

RequestCorrelIDToCorrelID
JMSCorrelationID est copié dans la zone JMSCorrelationID.

Ressources Java EE

Plusieurs ressources Java EE sont créées lors du déploiement d'une importation JMS MQ dans un environnement Java EE.

Paramètres

Fabrique de connexions MQ
Utilisée par les clients pour créer une connexion au fournisseur JMS MQ.

Fabrique de connexions de réponse
Utilisée par l'environnement d'exécution JMS MQ SCA lorsque la destination d'envoi se trouve sur un gestionnaire de files d'attente différent de celui de la destination de réception.

Spécification d'activation
Une spécification d'activation MQ JMS est associée à un ou plusieurs beans gérés par message et offre la configuration permettant de recevoir des messages.

Destinations

- Destination d'envoi :
  - Importations : Destination à laquelle la demande ou le message sortant est envoyé.
  - Exportations : Emplacement auquel est envoyé le message de réponse, si cette valeur n'est pas remplacée par l'en-tête JMSReplyTo dans le message entrant.
- Destination de réception :
  - Importations : Emplacement auquel est envoyé le message de réponse ou entrant.
  - Exportations : Emplacement dans lequel le message entrant doit être placé.

En-têtes JMS :

Un message JMS contient deux types d'en-têtes : l'en-tête de système JMS et plusieurs propriétés JMS. Il est possible d'accéder aux deux types d'en-tête depuis un module de médiation dans l'objet SMO (Service Message Object) ou en utilisant l'API ContextService.

En-tête système JMS

L'en-tête système JMS est représenté dans l'objet SMO par l'élément JMSHeader qui contient toutes les zones généralement présentes dans un en-tête JMS. Bien qu'elles puissent être modifiées dans la médiation (ou ContextService), certaines zones d'en-tête système JMS définies dans l'objet SMO ne seront pas propagées dans le message JMS sortant puisqu'elles sont remplacées par des valeurs statiques ou système.

Les zones clés de l'en-tête système JMS qui peuvent être mises à jour dans une médiation (ou ContextService) sont :

- **JMSType** et **JMSCorrelationID** : valeurs des propriétés de l'en-tête du message prédéfinies spécifiques
- **JMSDeliveryMode** : valeurs du mode de livraison (persistant (par défaut) ou non persistant)
- **JMSPriority** : valeur de la priorité (0 à 9 ; la valeur par défaut est JMS_Default_Priority)

Propriétés JMS

Les propriétés JMS sont représentées dans l'objet SMO en tant qu'entrées dans la liste Propriétés. Les propriétés peuvent être ajoutées, mises à jour ou supprimées dans une médiation ou en utilisant l'API ContextService.

Elles peuvent également être définies de manière statique dans la liaison JMS. Les propriétés définies de manière statique remplacent les paramètres (portant le même nom) qui sont définis de manière dynamique.

Les propriétés utilisateur propagées à partir d'autres liaisons (par exemple, une liaison HTTP) correspondront à une sortie dans la liaison JMS comme les propriétés JMS.

Paramètres de propagation d'en-tête

La propagation des propriétés et des en-têtes système JMS depuis le message JMS entrant vers les composants en aval ou depuis les composants en amont vers le message JMS sortant peut être contrôlée par l'indicateur Propagate Protocol Header.

Lorsque Propagate Protocol Header est défini, les informations d'en-tête peuvent circuler vers le message ou vers le composant cible, comme indiqué dans la liste suivante :

- Requête d'exportation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.
- Réponse d'exportation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'exportation JMS.
- Requête d'importation JMS

  Toute zone d'en-tête JMS définie dans le service de contexte sera utilisée dans le message sortant, excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS. Toute propriété définie dans le service de contexte sera utilisée dans le message sortant excepté en cas de remplacement par les propriétés statiques définies sur la liaison d'importation JMS.
- Réponse d'importation JMS

  L'en-tête JMS reçu dans le message sera propagé aux composants cible via le service de contexte. Les propriétés JMS reçues dans le message seront propagées aux composants cible via le service de contexte.

Clients externes :

Le serveur peut échanger des messages (envoi et réception) avec des clients externes par le biais de liaisons JMS WebSphere MQ.

Un client externe (comme un portail Web ou un système d'informations d'entreprise) peut envoyer un message à un composant SCA dans l'application par le biais d'une exportation ou il peut être appelé par un composant SCA dans l'application par le biais d'une importation.

La liaison d'exportation JMS WebSphere MQ déploie les beans MDB (Message Driven Beans) pour écouter les demandes entrantes sur la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si l'application appelée fournit une réponse. Ainsi, un client externe est en mesure d'appeler des applications via la liaison d'exportation.

Les importations JMS WebSphere MQ JMS effectuent des liaisons avec des clients externes et peuvent ainsi leur envoyer des messages. Ces messages peuvent ou non exiger une réponse de la part du client externe.

Pour plus d'informations sur l'interaction avec des clients externes à l'aide de WebSphere MQ, reportez-vous au centre de documentation WebSphere MQ.

Identification et résolution des incidents liés aux liaisons JMS WebSphere MQ :

Vous pouvez diagnostiquer et résoudre des incidents survenant sur les liaisons JMS WebSphere MQ.

Exceptions liées à l'implémentation

En réponse à diverses conditions d'erreur, l'implémentation de l'importation et de l'exportation JMS MQ peut renvoyer deux types d'exception :

- Service : cette exception est renvoyée si l'erreur définie sur l'interface métier de service (type de port WSDL) s'est produite.
- Service Runtime : générée dans tous les autres cas. Dans la plupart des cas, l'exception cause contient l'exception d'origine (JMSException).

  Par exemple, une importation n'attend qu'un seul message de réponse pour chaque message de demande. Si plusieurs réponses sont fournies, ou si une réponse tardive (une pour laquelle la réponse SCA a expiré) est fournie, une exception Service Runtime est générée. La transaction est annulée et le message de réponse est retiré de la file d'attente ou traité par le gestionnaire d'événements ayant échoué.

Messages SCA basés sur JMS WebSphere MQ qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si les messages SCA émis par le biais d'une interaction JMS WebSphere MQ échouent, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Or, si ces messages n'apparaissent pas, vérifiez que la valeur de la propriété du nombre maximal de nouvelles tentatives sur le port d'écoute sous-jacent est supérieure ou égale à **1**. Définir cette valeur sur **1** ou plus permet une interaction avec le gestionnaire d'événements ayant échoué au cours des appels SCA pour les liaisons JMS MQ.

Scénarios d'utilisation incorrecte : comparaison avec les liaisons WebSphere MQ

La liaison JMS WebSphere MQ est conçue pour l'interopérabilité avec les applications JMS déployées sur WebSphere MQ, où les messages affichés reposent sur un modèle de message JMS. Cependant, l'importation et l'exportationWebSphere MQ sont conçues essentiellement pour interopérer avec les applications WebSphere MQ natives et exposer le contenu intégral du corps de message WebSphere MQ aux médiations.

Dans les scénarios suivants, il est nécessaire d'utiliser la liaison JMSWebSphere MQ et non la liaison WebSphere MQ :

- Appel d'un bean géré par message (MDB) JMS à partir d'un module SCA, où le MDB est déployé sur le fournisseur JMS WebSphere MQ. Utilisez une importation JMS WebSphere MQ.
- Permettre au module SCA d'être appelé à partir d'un composant de servlet Java EE ou EJB par JMS. Utilisez une exportation JMS WebSphere MQ.
- Médiation du contenu d'un JMS MapMessage, transitant dans WebSphere MQ. Utilisez une exportation et une importation JMS WebSphere MQ conjointement avec le gestionnaire de données ou la liaison de données appropriés.

Dans certaines situations, la liaison WebSphere MQ et la liaison JMSWebSphere MQ peuvent interopérer. En particulier, si vous reliez des applications WebSphere MQ Java EE et non Java EE, utilisez une exportation WebSphere MQ et une importation JMS WebSphere MQ (ou vice-versa) conjointement avec les liaisons de données et/ou les modules de médiation appropriés (ou les deux).

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS est stocké dans le gestionnaire des événements ayant échoué par défaut pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Le gestionnaire de données étant appelé par la liaison de données, une exception de gestionnaire de données est générée dans une exception de liaison de données. Par conséquent, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Liaisons WebSphere MQ

La liaison WebSphere MQ assure une connectivité SCA (Service Component Architecture) avec les applications WebSphere MQ.

Utilisez les liaisons d'importation et d'exportation WebSphere MQ pour une intégration directe avec le système basé sur WebSphere MQ à partir de votre environnement serveur. Ainsi, il n'est plus nécessaire d'utiliser les fonctions de lien MQ ou client du bus d'intégration de services.

Lorsqu'un composant interagit avec un service WebSphere MQ par le biais d'une importation, la liaison d'importation WebSphere MQ utilise une file d'attente vers laquelle les données seront envoyées et une file d'attente où la réponse peut être reçue.

Lorsqu'un module SCA fournit un service aux clients WebSphere MQ, la liaison d'exportation WebSphere MQ utilise une file d'attente où la demande peut être reçue et la réponse envoyée. Le sélecteur de fonction sert à effectuer un mappage avec l'opération sur le composant cible à appeler.

La conversion des données utiles vers ou depuis un message MQ est effectuée par l'intermédiaire de la liaison de données ou du gestionnaire de données de corps MQ. La conversion des données d'en-tête vers et depuis un message MQ est effectuée par l'intermédiaire de la liaison de données d'en-tête MQ.

Pour plus d'informations sur les versions WebSphere MQ prises en charge, voir la page Web Configuration système requise détaillée.

Présentation des liaisons WebSphere MQ :

La liaison WebSphere MQ assure l'intégration avec les applications natives basées sur MQ.

Tâches d'administration WebSphere MQ

Avant d'exécuter une application contenant ce type de liaison, l'administrateur système WebSphere MQ est censé créer le gestionnaire de files d'attente WebSphere MQ sous jacent qui sera utilisé par les liaisons WebSphere MQ.

Tâches d'administration WebSphere

Vous devez spécifier comme propriété **Chemin d'accès aux bibliothèques natives**, de l'adaptateur de ressources MQ de Websphere, la version WebSphere MQ prise en charge par le serveur, puis redémarrer ce dernier. Cela garantit l'utilisation de bibliothèques d'une version de WebSphere MQ prise en charge. Pour plus de détails sur la configuration logicielle et matérielle requise, reportez-vous aux pages de support IBM..

Liaisons d'importation WebSphere MQ

La liaison d'importation WebSphere MQ permet aux composants au sein de votre module SCA de communiquer avec des services offerts par des applications externes basées sur WebSphere MQ. Vous devez utiliser une version prise en charge de WebSphere MQ. Pour plus de détails sur la configuration logicielle et matérielle requise, reportez-vous aux pages de support IBM..

L'interaction avec les systèmes WebSphere MQ externes comprend l'utilisation de file d'attentes pour l'envoi des requêtes et la réception des réponses.

Deux types descénarios d'utilisation pour la liaison d'importation WebSphere MQ sont pris en charge, en fonction du type d'opération appelée :

- Unidirectionnel : l'importation WebSphere MQ place un message sur la file d'attente configurée dans la zone **file d'attente de destination d'envoi** de la liaison d'importation. Rien n'est envoyé à la zone replyTo de l'en-tête MQMD.
- Bidirectionnel (demande-réponse) : l'importation WebSphere MQ place un message sur la file d'attente configurée dans la zone **file d'attente de destination d'envoi**

  La file d'attente receive est définie dans la zone d'en-tête MQMD replyTo. Un bean géré par message (MDB) est déployé pour écouter sur la file d'attente de réception et, dès qu'une réponse est reçue, le MDB transmet la réponse au composant.

La liaison d'importation peut être configurée (à l'aide de la zone **Schéma de corrélation de réponse**) pour faire en sorte que l'ID de corrélation de message de réponse soit copié à partir de l'ID de message de demande (valeur par défaut) ou à partir de l'ID de corrélation de message de demande.

Il est important de noter que WebSphere MQ est une liaison asynchrone. Si un composant appelant appelle une importation WebSphere MQ de manière synchrone (pour une opération bidirectionnelle), le composant appelant est bloqué jusqu'à ce que la réponse soit renvoyée par le service WebSphere MQ.

La figure 39, à la page 143 montre comment l'importation est liée au service externe.

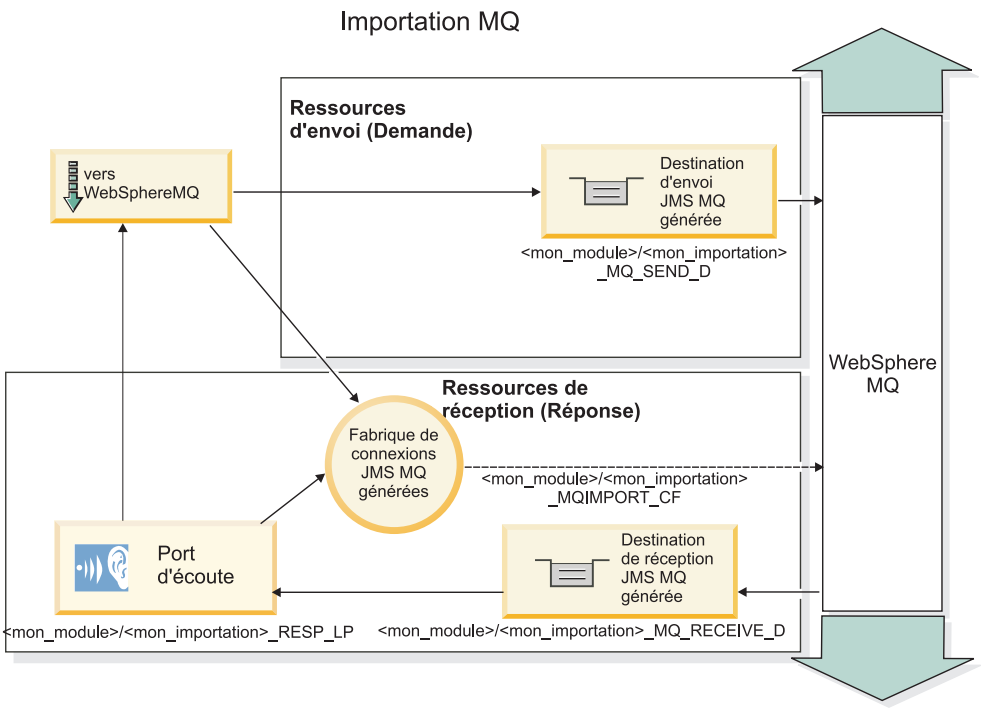


Figure 84. Ressources de liaison d'importation WebSphere MQ

Liaisons d'exportation WebSphere MQ

La liaison d'exportation WebSphere MQ offre les moyens aux modules SCA de fournir des services aux applications externes basées sur WebSphere MQ.

Un bean MDB est déployé pour écouter les demandes parvenant à la **file d'attente de destination de réception** spécifiée dans la liaison d'exportation. La file d'attente spécifiée dans la zone **file d'attente de destination d'envoi** est utilisée pour envoyer la réponse à la demande entrante si le composant appelé fournit une réponse. La file d'attente spécifiée dans la zone replyTo du message de réponse remplace la file d'attente spécifiée dans la zone **file d'attente de destination d'envoi**.

La figure 40, à la page 144 illustre la manière dont le demandeur externe est lié à l'exportation.

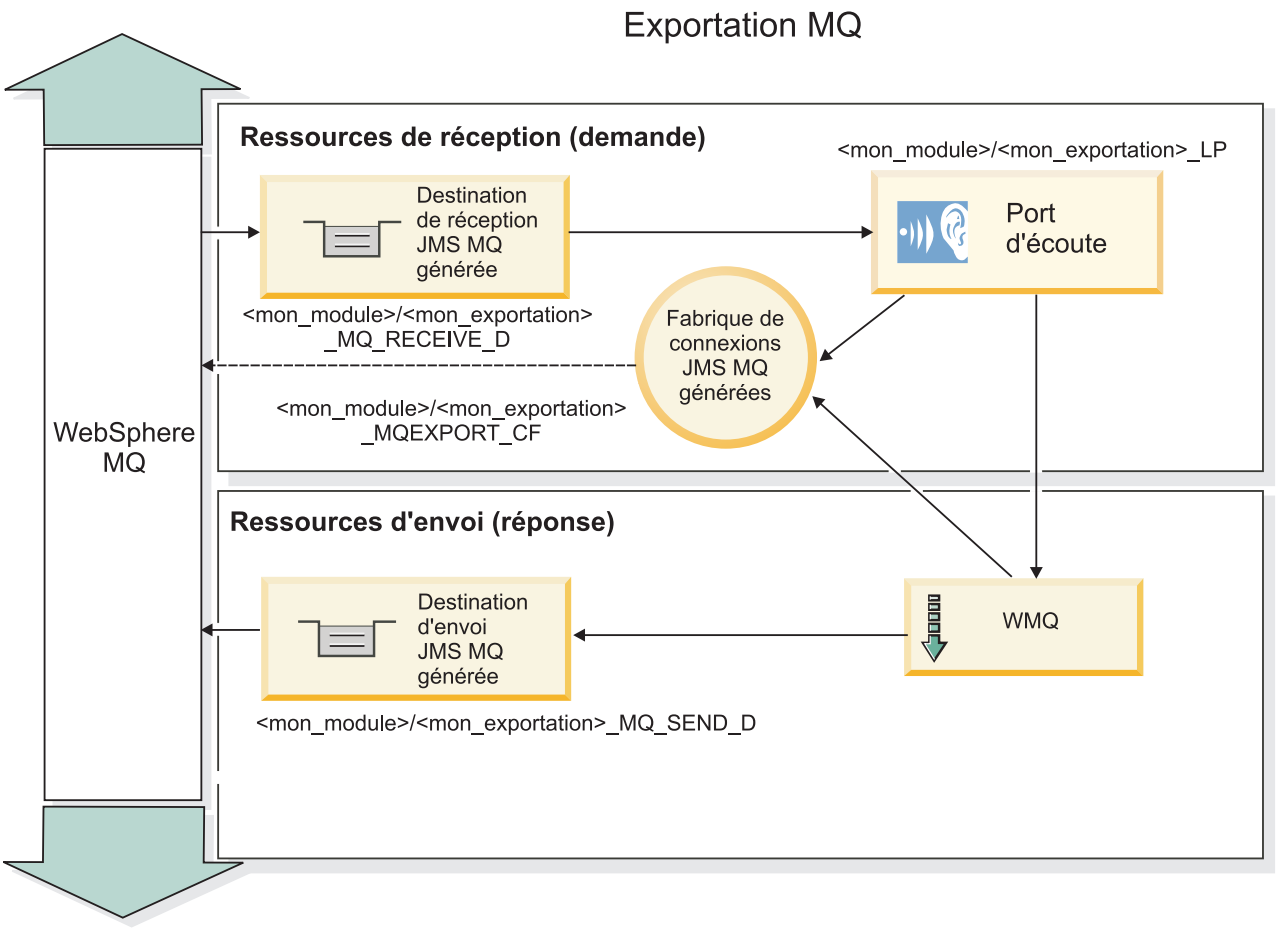


Figure 85. Ressources de liaison d'exportation WebSphere MQ

Remarque : La figure 39, à la page 143 et la figure 40, à la page 144 illustrent comment une application d'une version précédente de IBM Business Process Manager est associée à un service externe. Pour les applications développées pour IBM Business Process Manager, version 7 ou ultérieure, la spécification d'activation est utilisée à la place du port de l'écouteur et de la fabrique de connexions.

Principales fonctionnalités d'une liaison WebSphere MQ :

Les fonctionnalités essentielles d'une liaison WebSphere MQ incluent les en-têtes, les artefacts Java EE et les ressources Java EE créées.

Schémas de corrélation

Une application de demande/réponse WebSphere MQ peut utiliser différentes méthodes de corrélation entre les messages de réponse et les requêtes, qui reposent sur les zone MessageID et CorrelID du descripteur MQMD. Dans la plupart des cas, le demandeur laisse le gestionnaire de file d'attente sélectionner la valeur MessageID et s'attend à ce que l'application qui répond la copie dans la zone CorrelID de la réponse. En général, le demandeur et l'application qui répond savent implicitement quelle méthode de corrélation est utilisée. Parfois, l'application qui répond satisfait plusieurs indicateurs de la zone Report de la demande qui décrivent la méthode de traitement des zones.

Les liaisons d'exportation des messages WebSphere MQ peuvent être configurées à l'aide des options suivantes :

Options MsgId de la réponse :

New MsgID
Permet à un gestionnaire de file d'attente de sélectionner un MsgId pour la réponse (par défaut).

Copy from Request MsgID
Copie de la zone MsgId depuis la zone MsgId de la demande.

Copy from SCA message
Indique que le MsgId est celui des en-têtes de WebSphere MQ dans le message de réponse SCA, ou bien, en l'absence de valeur, c'est le gestionnaire de file d'attente qui définit un nouvel Id.

As Report Options
La zone Report du descripteur MQMD de la demande est analysée pour déterminer la manière de traiter MsgId. Les options MQRO_NEW_MSG_ID et MQRO_PASS_MSG_ID sont prises en charge et fonctionnent comme New MsgId et Copy from Request MsgID.

Options CorrelId de la réponse :

Copy from Request MsgID
Copie de la zone CorrelId depuis la zone MsgId de la demande (par défaut).

Copy from Request CorrelID
Copie de la zone CorrelId depuis la zone CorrelId de la demande.

Copy from SCA message
Indique que CorrelId est celui des en-têtes WebSphere MQ du message de réponse SCA, ou bien, en l'absence de valeur, cette zone reste vide.

As Report Options
La zone Report du descripteur MQMD de la demande est analysée pour déterminer la manière de traiter CorrelId. Les options MQRO_COPY_MSG_ID_TO_CORREL_ID et MQRO_PASS_CORREL_ID sont prises en charge et fonctionnent comme Copy from Request MsgID et Copy from Request CorrelID.

Les liaisons d'exportation des messages WebSphere MQ peuvent être configurées à l'aide des options suivantes :

Options MsgId de la demande :

New MsgID
Permet à un gestionnaire de file d'attente de sélectionner un MsgId pour la demande (par défaut).

Copy from SCA message
Indique que le MsgId est celui des en-têtes de WebSphere MQ dans le message de requête SCA, ou bien, en l'absence de valeur, permet au gestionnaire de file d'attente de définir un nouvel Id.

Options de corrélation de réponse :

Response has CorrelID copied from MsgId
Dans le message de réponse, la zone CorrelId doit être définie en fonction de la valeur MsgId de la requête (par défaut).

Response has MsgID copied from MsgId
Dans le message de réponse, la zone MsgId doit être définie en fonction de la valeur MsgId de la requête.

Response has CorrelID copied from CorrelId
Dans le message de réponse, la zone CorrelId doit être définie en fonction de la valeur CorrelId de la requête.

Ressources Java EE

Plusieurs ressources Java EE sont créées lorsqu'une liaison WebSphere MQ est déployée dans un environnement Java EE.

Paramètres

Fabrique de connexions MQ
Utilisée par les clients pour créer une connexion au fournisseur WebSphere MQ.

Fabrique de connexions de réponse
Utilisée par l'environnement d'exécution JMS MQ SCA lorsque la destination d'envoi se trouve sur un gestionnaire de files d'attente différent de celui de la destination d'envoi.

Spécification d'activation
Une spécification d'activation MQ JMS est associée à un ou plusieurs beans gérés par message et offre la configuration permettant de recevoir des messages.

Destinations

- Destination d'envoi : Destination à laquelle est envoyé le message de demande ou sortant (importation) ou le message de réponse (exportation), si la valeur n'est pas remplacée par la zone d'en-tête ReplyTo du MQMD du message entrant.
- Destination de réception : Emplacement dans lequel le message de réponse/demande ou le message entrant doit être placé.

En-têtes WebSphere MQ :

Les en-têtes WebSphere MQ incorporent certaines conventions pour la conversion aux messages de l'architecture SCA (Service Component Architecture).

Les messages WebSphere MQ se composent d'un en-tête système (le MQMD), de zéro ou plusieurs autres en-têtes MQ (système ou personnalisés) et d'un corps. S'il contient plusieurs en-têtes, leur ordre est important.

Chaque en-tête contient des informations décrivant la structure de l'en-tête suivant. Le MQMD décrit le premier en-tête.

Mode d'analyse des en-têtes MQ

Une liaison de données d'en-tête MQ est utilisée pour analyser les en-têtes MQ. Les en-têtes suivants sont automatiquement pris en charge :

- MQRFH
- MQRFH2
- MQCIH
- MQIIH

Les en-têtes qui commencent par **MQH** sont traités de façon différente. Certains champs de l'en-tête ne sont pas analysés ; ils persistent sous la forme d'octets non analysés.

Pour d'autres en-têtes MQ, vous pouvez écrire des liaisons de données d'en-tête MQ personnalisées afin d'analyser ces en-têtes.

Mode d'accès aux en-têtes MQ

Vous pouvez accéder aux en-têtes MQ de l'une des manières suivantes :

- Via l'objet de message de service (SMO) dans une médiation

- Via l'API ContextService

Les en-têtes MQ sont représentés en interne avec l'élément SMO MQHeader. MQHeader est un conteneur de données d'en-tête qui étend MQControl, mais qui contient un élément de valeur anyType. Il inclut MQMD, MQControl (informations de contrôle du corps de message MQ), ainsi qu'une liste d'autres en-têtes MQ.

- MQMD représente le contenu de la description du message WebSphere MQ, excepté pour les informations déterminant la structure et le codage du corps.
- MQControl contient des informations déterminant la structure et le codage d'un corps de message.
- MQHeaders contient une liste d'objets MQHeader.

La chaîne MQ n'est pas affectée de sorte que, dans le SMO, chaque en-tête MQ inclut ses propres informations de contrôle (CCSID, codage et format). Les en-têtes peuvent être facilement ajoutés ou supprimés, sans modifier d'autres données d'en-tête.

Définition de champs dans le MQMD

Vous pouvez mettre à jour le MQMD à l'aide de l'API Context ou via l'objet de message de service (SMO) dans une médiation. Les champs suivants sont automatiquement propagés vers le message MQ sortant :

- Encoding
- CodedCharacterSet
- Format
- Report
- Expiry
- Feedback
- Priority
- Persistence
- CorrelId
- MsgFlags

Configurez la liaison MQ sur une importation ou une exportation pour propager les propriétés suivantes vers le message MQ sortant :

MsgID
Définissez **Request Message ID** à `copy from SCA message`.

MsgType
Décochez la case **Set message type to MQMT_DATAGRAM or MQMT_REQUEST for request-response operation**.

ReplyToQ
Décochez la case **Override reply to queue of request message**.

ReplyToQMgr
Décochez la case **Override reply to queue of request message**.

A partir de la version 7.0, les champsde contexte peuvent être remplacés à l'aide d'une propriété personnalisée dans la définition de la destination JNDI. Définissez la propriété personnalisée MDCTX avec la valeur `SET_IDENTITY_CONTEXT` dans la destination d'envoi pour propager les champssuivants vers le message MQ sortant :

- UserIdentifier
- AppIdentityData

Définissez la propriété personnalisée MDCTX avec la valeur SET_ALL_CONTEXT dans la destination d'envoi pour propager les propriétés suivantes vers le message MQ sortant :

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Certains champs ne sont pas propagés vers le message MQ sortant. Les champs suivants sont remplacés lors de l'envoi du message :

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

Ajout statique de MQCIH dans une liaison WebSphere MQ :

IBM Business Process Manager prend en charge l'ajout des informations d'en-tête MQCIH statiquement sans utiliser de module de médiation.

Il existe différentes façons d'ajouter des informations d'en-tête MQCIH à un message (par exemple, en utilisant la primitive de médiation Configurateur d'en-tête). Il peut être utile d'ajouter des informations d'en-tête statiquement, sans utiliser de module de médiation supplémentaire. Les informations d'en-tête statique, y compris le nom du programme CICS, l'ID de transaction et d'autres détails d'en-tête de format de données peuvent être définis et ajoutés dans le cadre de la liaison WebSphere MQ.

WebSphere MQ, MQ CICS Bridge, et CICS doivent être configurés de telle sorte que les informations d'en-tête MQCIH soient ajoutées statiquement.

Vous pouvez utiliser Integration Designer de façon à configurer l'importation WebSphere MQ avec des valeurs statiques qui sont nécessaires pour les informations d'en-tête MQCIH.

Lorsqu'un message arrive et est traité par l'importation WebSphere MQ, une vérification est effectuée pour vérifier que les informations d'en-tête MQCIH figurent déjà dans le message. Si MQCIH est présent, les valeurs statiques définies dans l'importation WebSphere MQ sont utilisées pour remplacer les valeurs dynamiques correspondantes du message. Si MQCIH est absent, il en est créé un dans le message et les valeurs statiques définies dans l'importation WebSphere MQ sont ajoutées.

Les valeurs statiques définies dans l'importation WebSphere MQ sont spécifiques à une méthode. Vous pouvez spécifier différentes valeurs MQCIH statiques pour différentes méthodes au sein d'une même importation WebSphere MQ.

Cette fonction n'est pas utilisée pour donner des valeurs par défaut si MQCIH ne contient pas les informations d'en-tête spécifiques car une valeur statique définie dans l'importation WebSphere MQ remplacera la valeur correspondante fournie par le message entrant.

Clients externes :

IBM Business Process Manager peut échanger des messages (envoi et réception) avec des clients externes par le biais de liaisons WebSphere MQ.

Un client externe (comme un portail Web ou un système d'informations d'entreprise) peut envoyer un message à un composant SCA dans l'application par le biais d'une exportation ou il peut être appelé par un composant SCA dans l'application par le biais d'une importation.

La liaison d'exportation WebSphere MQ déploie les beans MDB (Message Driven Beans) pour écouter les demandes entrantes sur la destination receive spécifiée dans la liaison d'exportation. La destination spécifiée dans la zone send est utilisée pour envoyer la réponse à la demande entrante si l'application appelée fournit une réponse. Ainsi, un client externe est en mesure d'appeler des applications par l'intermédiaire d'une liaison d'exportation.

Les importations JMS WebSphere MQ effectuent des liaisons avec des clients externes et peuvent ainsi leur envoyer des messages. Ces messages peuvent ou non exiger une réponse de la part du client externe.

Pour plus d'informations sur l'interaction avec des clients externes à l'aide de WebSphere MQ, reportez-vous au centre de documentation WebSphere MQ.

Identification des incidents liés aux liaisons WebSphere MQ :

Diagnostic et correction des incidents et des erreurs liés aux liaisons WebSphere MQ.

Conditions d'erreur principales

Les principales conditions d'erreur liées aux liaisons WebSphere MQ peuvent être identifiées via la sémantique transactionnelle, la configuration de WebSphere MQ ou par référence au fonctionnement sur d'autres composants. Les causes premières d'incident peuvent être :

- Erreur de connexion au gestionnaire de file d'attente ou à la file d'attente WebSphere MQ.

  Une erreur de connexion à WebSphere MQ destinée à recevoir des messages entraîne l'échec de démarrage du port d'écoute MDB. Cette situation sera enregistrée dans la journal WebSphere Application Server. Des messages persistants resteront dans la file d'attente de WebSphere MQ jusqu'à ce qu'ils soient récupérés (ou que WebSphere MQ les fasse expirer).

  Une erreur de connexion à WebSphere MQ destinée à envoyer des messages sortants entraîne l'annulation de la transaction qui contrôle l'envoi.
- Erreur d'analyse d'un message entrant ou de construction d'un message sortant.

  Une erreur de liaison des données ou du gestionnaire de données entraîne l'annulation de la transaction qui contrôle le travail.
- Erreur d'envoi du message sortant.

  Un échec d'envoi d'un message entraîne l'annulation de la transaction en question.
- Messages de réponse multiples ou inattendus.

  L'importation n'attend qu'un seul message de réponse pour chaque message de demande. Si plusieurs réponses sont fournies, ou si une réponse tardive (une pour laquelle la réponse SCA a expiré) est fournie, une exception Service Runtime est générée. La transaction est annulée et le message de réponse est retiré de la file d'attente ou traité par le gestionnaire d'événements ayant échoué.

Scénarios d'utilisation incorrecte : comparaison avec les liaisons WebSphere MQ

L'importation et l'exportation WebSphere MQ sont conçues essentiellement pour interopérer avec les applications natives WebSphere MQ et exposer le contenu intégral du corps de message WebSphere MQ aux médiations. Toutefois, la liaison JMS WebSphere MQ est conçue pour l'interopérabilité avec les applications JMS déployées sur WebSphere MQ, où les messages affichés reposent sur un modèle de message JMS.

Dans les scénarios suivants, il est nécessaire d'utiliser la liaison JMSWebSphere MQ et non la liaison WebSphere MQ :

- Appel d'un bean géré par message (MDB) JMS à partir d'un module SCA, où le MDB est déployé sur le fournisseur JMS WebSphere MQ. Utilisez une importation JMS WebSphere MQ.
- Permettre au module SCA d'être appelé à partir d'un composant de servlet Java EE ou EJB par JMS. Utilisez une exportation JMS WebSphere MQ.
- Médiation du contenu d'un JMS MapMessage, transitant dans WebSphere MQ. Utilisez une exportation et une importation JMS WebSphere MQ conjointement avec la liaison de données appropriée.

Dans certaines situations, la liaison WebSphere MQ et la liaison JMSWebSphere MQ peuvent interopérer. En particulier, si vous reliez des applications WebSphere MQ Java EE et non Java EE, utilisez une exportation WebSphere MQ et une importation JMS WebSphere MQ (ou vice-versa) conjointement avec les liaisons de données et/ou les modules de médiation appropriés (ou les deux).

Messages non délivrés

Si WebSphere MQ ne parvient pas envoyer un message à la destination prévue (en règle générale, suite à des erreurs de configuration), il envoie le message à une file d'attente de rebut.

Dans ce cas, il ajoute un en-tête de non-distribution au début du corps de message. Ce dernier indique les raisons de l'erreur, la destination d'origine, ainsi que d'autres informations.

Messages SCA basés sur MQ qui n'apparaissent pas dans le gestionnaire des événements ayant échoué

Si des messages SCA ont été générés en raison d'un incident d'interaction WebSphere MQ, vous devriez les retrouver dans le gestionnaire des événements ayant échoué. Si ces messages ne s'affichent pas dans le gestionnaire des événements ayant échoué, vérifiez que la destination WebSphere MQ sous-jacente possède une valeur du nombre maximal de livraisons ayant échoué supérieure à 1. Spécifier une valeur supérieure ou égale à 2 permet une interaction avec le gestionnaire d'événements ayant échoué au cours des appels SCA pour les liaisons WebSphere MQ.

Les événements MQ ayant échoué sont lus à nouveau sur le gestionnaire de files d'attente incorrect.

Quand une fabrique de connexions prédéfinie est utilisée pour les connexions sortantes, les propriétés de connexion doivent correspondre à celles définies dans la spécification d'activation employée pour les connexions entrantes.

La fabrique de connexions prédéfinies est utilisée pour créer une connexion quand un événement ayant échoué est relu. Elle doit donc être configurée pour utiliser le même gestionnaire de files d'attente duquel le message a été reçu à l'origine.

Gestion des exceptions :

La configuration de la liaison détermine la manière dont sont gérées les exceptions qui sont émises par les gestionnaires de données ou les liaisons de données. En outre, la nature du flux de médiation détermine le comportement du système lorsqu'une exception de ce type est générée.

Différents incidents peuvent se produire lorsqu'un gestionnaire de données ou une liaison de données est appelé(e) par votre liaison. Par exemple, un gestionnaire de données peut recevoir un message dont le contenu est endommagé ou il peut essayer de lire un message dont le format est incorrect.

Le traitement d'une telle exception par votre liaison est déterminé par la manière dont vous implémentez le gestionnaire de données ou la liaison de données. Il est ainsi recommandé de configurer votre liaison de données afin qu'elle génère une exception de type **DataBindingException**.

La situation est similaire pour un gestionnaire de données. Dès lors que le gestionnaire de données est appelé par la liaison de données, toute exception de gestionnaire de données est encapsulée dans une exception de liaison de données. Ainsi, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Lorsqu'une exception d'exécution, y compris une exception **DataBindingException** est émise :

- Si le flux de médiation est configuré pour être transactionnel, le message JMS est stocké dans le gestionnaire des événements ayant échoué par défaut pour une relecture manuelle ou une suppression.

  Remarque : Vous pouvez modifier le mode de récupération sur la liaison afin que le message soit annulé plutôt que stocké dans le gestionnaire des événements ayant échoué.
- Si le flux de médiation n'est pas de type transactionnel, l'exception est consignée et le message est perdu.

La situation est similaire pour un gestionnaire de données. Le gestionnaire de données étant appelé par la liaison de données, une exception de gestionnaire de données est générée dans une exception de liaison de données. Par conséquent, une exception **DataHandlerException** vous est signalée sous la forme d'une exception **DataBindingException**.

Limitations des liaisons

L'utilisation des liaisons connaît certaines limitations qui sont répertoriées ici.

Limitations de la liaison MQ :

L'utilisation de la liaison MQ connaît certaines limitations qui sont répertoriées ici.

Pas de distribution de messages par publication/abonnement

La méthode de distribution des messages par publication/abonnement n'est pas prise en charge par la liaison MQ, bien que WMQ la prenne en charge. Toutefois, la liaison JMS MQ prend en charge cette méthode de distribution.

Files d'attentes de réception partagées

Plusieurs liaisons d'exportation et d'importation WebSphere MQ attendent que tous les messages présents dans la file d'attente de réception configurée soient destinés à l'exportation ou l'importation. Les liaisons d'importation et d'exportation doivent être configurées en prenant en compte ce qui suit :

- Chaque importation MQ doit avoir une file d'attente de réception distincte, car les liaisons d'importation MQ supposent que tous les messages dans la file d'attente de réception sont des réponses à des demandes qu'elles ont envoyées. Si la file d'attente de réception est partagée par plusieurs importations, les réponses peuvent être reçues par la mauvaise importation et ne seront pas mises en corrélation avec le message de demande d'origine.
- Chaque exportation MQ doit avoir une file d'attente de réception distincte, sinon vous ne pouvez pas prévoir quelle exportation recevra un message de demande déterminé.
- Les importations et les exportations MQ peuvent désigner la même file d'attente d'envoi.

Limitations des liaisons JMS, JMS MQ et JMS génériques :

Les liaisons JMS et JMS MQ connaissent certaines limitations.

Implications de la génération de liaisons par défaut

Les limitations de l'utilisation de liaisons JMS, JMS MQ et JMS génériques sont présentées dans les sections suivantes :

- Implications de la génération de liaisons par défaut

- Schéma de corrélation des réponses
- Support bidirectionnel

Lorsque vous générez une liaison, plusieurs zones sont renseignées automatiquement par défaut, si vous ne choisissez pas d'entrer les valeurs vous-même. Par exemple, un nom de fabrique de connexions est créé automatiquement. Si vous savez que vous placerez votre application sur un serveur et que vous y accéderez à distance à l'aide d'un client, vous devez, lors de la création de la liaison, entrez des noms JNDI plutôt que d'utiliser les valeurs par défaut car vous souhaiterez certainement contrôler ces valeurs via la console d'administration lors de la phase d'exécution.

Toutefois, si vous avez accepté les valeurs par défaut, mais que vous découvrez par la suite que vous ne pouvez pas accéder à votre application à partir d'un client éloigné, vous pouvez utiliser la console d'administration pour définir explicitement la valeur de la fabrique de connexions. Recherchez la zone des noeuds finaux du fournisseur dans les paramètres de la fabrique de connexions et ajoutez une valeur telle que la suivante : <nomhôte_serveur>:7276 (si vous utilisez le numéro de port par défaut).

Schéma de corrélation des réponses

Si vous utilisez le schéma de corrélation des réponses CorrelationId à CorrelationId, permettant de corréler les messages dans une opération demande/réponse, le message doit contenir un ID corrélation dynamique.

Pour créer un ID corrélation dynamique dans un module de médiation à l'aide de l'éditeur de flux de médiation, ajoutez une primitive de médiation de Mappage avant l'importation avec la liaison JMS. Ouvrez l'éditeur de mappage. Les en-têtes connus de l'architecture des composants de service seront disponibles dans le message cible. Faites glisser-déposer une zone contenant un ID unique dans le message dans l'ID corrélation de l'en-tête JMS du message cible.

Support bidirectionnel

Seuls les caractères ASCII sont pris en charge pour les noms JNDI (Java Naming and Directory Interface) lors de l'exécution.

Files d'attentes de réception partagées

Plusieurs liaisons d'exportation et d'importation attendent que tous les messages présents dans la file d'attente de réception configurée soient destinés à l'exportation ou l'importation. Les liaisons d'importation et d'exportation doivent être configurées en prenant en compte ce qui suit :

- Chaque liaison d'importation doit avoir une file d'attente de réception distincte, car ces liaisons supposent que tous les messages dans la file d'attente de réception sont des réponses à des demandes qu'elles ont envoyées. Si la file d'attente de réception est partagée par plusieurs importations, les réponses peuvent être reçues par la mauvaise importation et ne seront pas mises en corrélation avec le message de demande d'origine.
- Chaque exportation doit avoir une file d'attente de réception distincte, sinon vous ne pouvez pas prévoir quelle exportation recevra un message de demande déterminé.
- Les importations et les exportations peuvent désigner la même file d'attente d'envoi.

Objets métier

Les développeurs de logiciels ont mis au point plusieurs modèles et infrastructures de programmation dans lesquels les *objets métier* offrent une représentation naturelle des données métier pour le traitement des applications.

D'une manière générale, ces objets métier :

- Sont définis conformément aux normes du secteur informatique

- Mappent de façon transparente des données aux tables de base de données ou aux systèmes d'information d'entreprise.
- Prennent en charge des protocoles d'appels
- Fournissent la base du modèle de programmation des données pour la programmation des applications

Process Designer et Integration Designer fournissent aux développeurs un modèle d'objet métier commun pour la représentation des divers types d'entités commerciales issues de domaines différents.

Dans Process Designer, les objets métier reposent sur une représentation de type de données. Des objets métier de base (types de variable) sont fournis dans des kits d'outils système. Vous pouvez créer des types de variable personnalisés appelés "objets métier personnalisés". Pour plus d'informations, reportez-vous à la rubrique Objets métier et variables.

Dans Integration Designer, disponible avec IBM Business Process Manager Advanced, les objets métier peuvent représenter des constructions XSD plus complexes. Dans Integration Designer, les objets métier ont une affinité étroite avec les schémas XML. Pour plus d'informations sur les éléments à prendre en considération lors de l'intégration des objets métier définis dans Integration Designer à des objets métier définis dans Process Designer, reportez-vous aux rubriques Library mirroring et Constructions XML non prises en charge.

Lors de la phase de développement dans Integration Designer, le modèle d'objet métier permet aux développeurs de définir des objets métier sous la forme de définitions de schémas XML. Lors de la phase d'exécution, les données métier établies par les définitions de schéma XML sont représentées sous forme d'objets métier Java. Dans ce modèle, les objets métier s'appuient assez librement sur des avant projets de la spécification SDO (Service Data Object) et offrent un jeu complet d'interfaces d'applications des modèles de programmation requis pour la manipulation des données de gestion. Les sous-rubriques suivantes expliquent plus en détail comment vous pouvez utiliser des objets métier avec Integration Designer.

Définir des objets métier

La définition d'objets métier s'effectue à l'aide de l'éditeur livré avec Integration Designer. Cet éditeur stocke les objets métier sous forme de définitions de schémas XML.

L'utilisation d'un schéma XML pour définir des objets métier présente plusieurs avantages :

- Les schémas XML fournissent un modèle standard de définition des données et une base pour l'interopérabilité entre des systèmes et des applications disparates et hétérogènes. Ils s'utilisent conjointement au langage WSDL (Web Services Description Language) pour fournir des contrats d'interface standard entre les composants, les applications et les systèmes.
- Les modèles de définition de données qu'ils produisent sont riches en informations permettant de représenter les données métier. Entre autres fonctionnalités intéressantes, ces modèles incluent des types complexes, des types simples, des types définis par l'utilisateur, l'héritage des types et la cardinalité.
- Les objets métier peuvent être définis aussi bien par les interfaces métier et les données définies dans le langage WSDL (Web Services Description Language) que par des schémas XML émanant d'organismes édicteurs de normes sectorielles ou provenant d'autres systèmes ou d'autres applications. Integration Designer peut importer directement ces objets métier.

Integration Designer permet également de détecter les données métier présentes dans des bases de données et dans des systèmes d'information d'entreprise et de générer les schémas XML de ces données métier. L'on désigne fréquemment les objets métier générés de la sorte sous le nom d'*objets métier propres à une application* car ils imitent la structure des données métier définies dans le système d'information d'entreprise.

Lorsqu'un processus manipule des données provenant d'un grand nombre de systèmes d'information différents, il peut être intéressant de transformer les représentations disparates des données métier (par exemple, CustomerEIS1 et CustomerEIS2 ou OrderEIS1 et OrderEIS2) en une représentation canonique unique (par exemple, Customer ou Order). L'on désigne souvent la représentation canonique sous le nom d'*objet métier générique*.

Les définitions d'objets métier, et c'est particulièrement vrai des objets métier génériques, sont souvent utilisées par plus d'une application. Pour permettre cette réutilisation, Integration Designer autorise la création d'objets métier dans des bandothèques qui peuvent dès lors être associées à plusieurs modules d'application.

Le langage WSDL (Web Services Description Language) définit les contrats des services fournis et utilisés par un module d'application SCA (Service Component Architecture), ainsi que les contrats permettant de créer les composants d'un module d'application. Dans un contrat, un WSDL peut représenter les opérations et les objets métier (définis par un schéma XML pour représenter les données métier).

Utilisation des objets métier

SCA (Service Component Architecture) fournit le cadre de travail permettant de définir un module d'application, les services que fournit ce module, ceux qu'il utilise et la composition des composants fournissant la logique métier du module. Les objets métier jouent un rôle important dans l'application, en définissant les données métier servant à décrire les contrats du service et de ses composants ainsi que les données métier manipulées par ces composants.

Le schéma suivant montre un module d'application SCA avec l'indication d'un grand nombre d'endroits où le développeur exploite les objets métier.

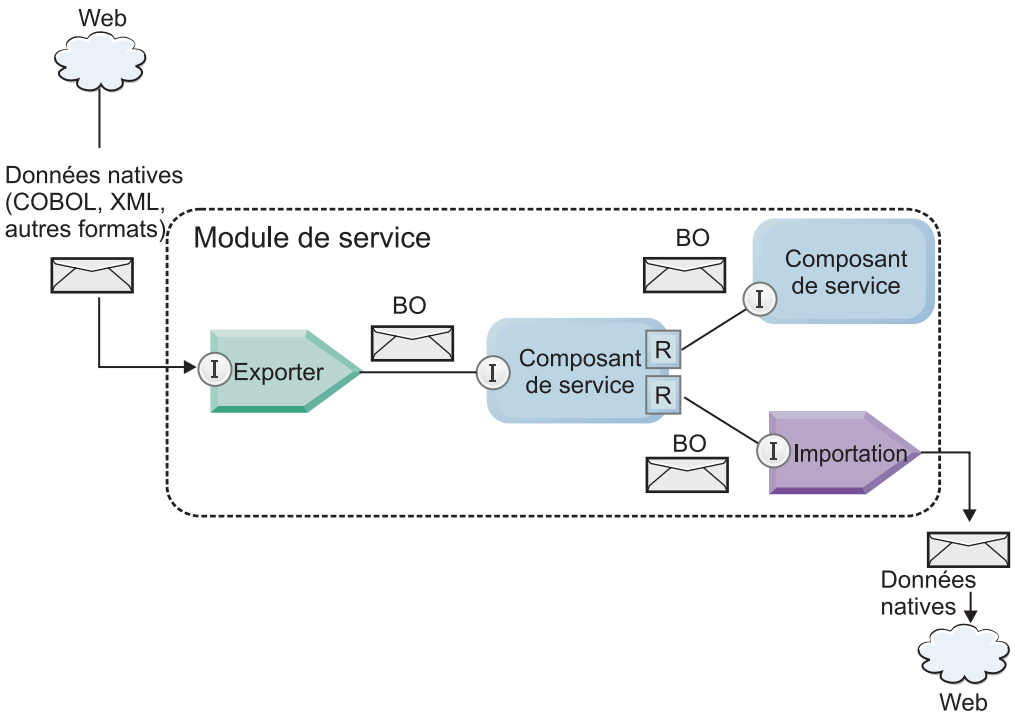


Figure 86. Les objets métier représentent les données qui circulent entre les services dans une application.

Remarque : Nous allons décrire l'utilisation des objets métier par les modules d'applications SCA. Si vous utilisez des interfaces Java, les modules d'applications SCA pourront également traiter les objets Java.

Modèle de programmation des objets métier

Le modèle de programmation des objets métier est constitué d'un ensemble d'interfaces Java représentant :

- la définition et les données d'instance des 'objets métier
- un ensemble de services prenant en charge les opérations sur les objets métier

Les définitions de type d'objet métier sont représentées par les interfaces commonj.sdo.Type et commonj.sdo.Property. Le modèle de programmation fournit un ensemble de règles pour le mappage vers l'interface Type des types complexes de schémas XML et pour le mappage de chacun des éléments de la définition de type complexe vers l'interface Property.

Les instances d'objets métier sont représentées par l'interface commonj.sdo.DataObject. Le modèle de programmation des objets métier est sans types, ce qui signifie que la même interface commonj.sdo.DataObject peut être utilisée pour représenter plusieurs définitions d'objets métier différentes, comme Customer et Order, par exemple. Savoir quelles propriétés seront définies et extraites de chaque objet métier dépend des informations de type définies dans le schéma XML associé à l'objet métier.

Le comportement du modèle de programmation d'objet métier repose sur la spécification Service Data Object 2.1. Pour plus d'informations, voir la spécification SDO 2.1 for Java, les tutoriels et les javadocs sur le site Web : http://www.oasis-opencsa.org/.

Les services d'objets métier prennent en charge diverses opérations liées au cycle de vie (création, égalité, analyse syntaxique, sérialisation, etc.) des objets métier.

Pour des informations détaillées sur le modèle de programmation des objets métier, voir Programmation à l'aide de services d'objet métier ainsi que la documentation sur les interfaces API et SPI générées relative aux objets métier.

Liaisons, liaisons de données et gestionnaires de données

Comme le montre la figure 41, à la page 154, les données métier servant à appeler des services fournis par les modules d'applications SCA sont transformées en objets métier pour permettre aux composants SCA de manipuler les données métier. De la même manière les objets métier manipulés par les composants SCA sont convertis dans le format de données requis par les services externes.

Dans certains cas, comme la liaison à un service Web, la liaison servant à exporter et importer des services transforme automatiquement les données dans le format approprié. Dans d'autres cas, comme dans la liaison JMS, les développeurs peuvent fournir une liaison de données ou un gestionnaire de données qui convertissent les formats non natifs en objets métier représentés par l'interface DataObject.

Pour plus d'informations sur le développement de liaisons et de gestionnaires de données, voir «Gestionnaires de données», à la page 58 et «Liaisons de données», à la page 60.

Composants

Les composants SCA définissent leurs contrats de services de fourniture et d'utilisation à l'aide d'une combinaison de langage WSDL et de schémas XML. Les données métier que SCA transmet entre les composants sont représentées sous forme d'objets métier à l'aide de l'interface DataObject. SCA vérifie que ces types d'objets métier sont compatibles avec le contrat d'interface défini par le composant à appeler.

Les abstractions du modèle de programmation pour la manipulation des objets métier varient d'un composant à l'autre. Le composant POJO et la primitive Custom du composant de flux de médiation

fournissent une manipulation directe des objets métier en permettant la programmation Java directe à l'aide des interfaces de programmation et des services d'objets métier. La plupart des composants fournissent des abstractions générales pour la manipulation des objets métier tout en fournissant également des fragments de code Java pour la définition de comportements personnalisés dans les interfaces et les services d'objets métier.

Il est possible de transformer les objets métier à l'aide des composants Interface Flow Mediation et Business Object Map (Mappe d'objet métier) ou du composant de flux de médiation et de sa primitive Mappe XML. Ces possibilités de transformations sont utiles pour la conversion dans les deux sens d'objets métier spécifiques à une application en objets métier génériques.

Objets métier spéciaux

Les objets de message de service et les graphiques métier sont deux types spécialisés d'objets métier qui sont utilisés à des fins spécifiques.

Objet de message de service

Un objet de message de service (SMO) est un objet métier spécialisé qui est utilisé par les composants de flux de médiation pour représenter la collection des données associées à un appel de service.

Un SMO a une structure fixe de premier niveau constituée des en-têtes, du contexte, du corps du message et des éventuelles pièces jointes.

- Les en-têtes charrient les informations relatives à l'appel de service via un protocole ou une liaison particulière. Exemples : en-têtes SOAP ou JMS.
- Les données de contexte transportent les informations logiques supplémentaires associées à l'appel lorsque celui-ci est traité par le composant de flux de médiation. En principe, ces informations ne font pas partie des données d'application envoyées ou reçues par les clients.
- Le corps du SMO contient les données métier elles-mêmes, qui représentent sous la forme d'un objet métier standard le message central de l'application ou les données de l'appel.

Le SMO peut également transporter des données jointes dans le cas d'appels à des services Web utilisant SOAP avec des pièces jointes.

Les flux de médiation effectuent des tâches comme le routage des demandes et la transformation des données et le SMO unifie dans une même structure les en-têtes et le contenu du message.

Graphique métier

Un graphique métier est un objet métier spécial servant à permettre la synchronisation des données dans des scénarios d'intégration.

Prenons l'exemple de deux systèmes d'information d'entreprise qui ont une représentation d'une commande spécifique. Lorsque la commande est modifiée dans l'un des deux systèmes, un message peut être envoyé à l'autre système afin de synchroniser les données de la commande. Les graphiques métier se chargent d'expédier à l'autre système uniquement la portion de la commande qui a changé, en ajoutant des annotations définissant précisément la teneur des modifications.

Dans notre exemple, un graphique métier Order n'enverra à l'autre système que l'article de la commande qui a été supprimé ainsi que la date modifiée de l'expédition de la commande.

Dans Integration Designer, il est facile d'ajouter des graphiques métier aux objets métier existants. Ils sont généralement utilisés dans les scénarios dans lesquels des adaptateurs WebSphere sont utilisés et pour prendre en charge la migration des applications WebSphere InterChange Server.

Mode d'analyse syntaxique d'objet métier

Integration Designer fournit une propriété pour les modules et les bibliothèques, qui permet de configurer le mode d'analyse XML pour les objets métier : rapide ou lent.

- Si le mode doit être *rapide*, les flux des octets XML sont analysés rapidement pour la création de l'objet métier.
- Avec l'option *lent*, l'objet métier est créé normalement et l'analyse effective du flux des octets XML est différée, une analyse partielle ne s'effectuant que lors de l'accès aux propriétés de l'objet métier.

Dans les deux modes, les données non XML sont toujours analysées de manière rapide pour la création de l'objet métier.

Remarques relatives au choix du mode d'analyse syntaxique des objets métier

Le mode d'analyse syntaxique des objets métier détermine le mode d'analyse syntaxique des données XML. Un mode d'analyse syntaxique d'objet métier est défini dans un module ou une bibliothèque lors de sa création. Vous pouvez changer le mode d'analyse syntaxique du module ou de la bibliothèque, mais vous devez connaître les conséquences.

Le mode d'analyse d'objet métier est défini au niveau du module ou de la bibliothèque. Les modules créés dans une version IBM Integration Designer antérieure à la version 7 fonctionnent dans le mode d'analyse syntaxique systématique sans avoir à effectuer des modifications. Par défaut, les modules et les bibliothèques créés dans IBM Integration Designer version 7 et les versions suivantes sont affectés du mode d'analyse syntaxique le plus approprié en fonction de divers facteurs, tels que le mode d'analyse syntaxique des projets existants dans l'espace de travail ou du mode d'analyse syntaxique des projets dépendants ou d'autres projets dans la même solution, etc. Vous pouvez changer le mode d'analyse syntaxique du module ou de la bibliothèque en fonction de l'implémentation, mais vous devez connaître les conséquences.

Considérations

- Le mode d'analyse syntaxique lente des objets métier traite plus vite les données XML, mais il existe des différences de compatibilité que vous devez connaître entre le mode systématique et le mode lent avant de changer la configuration d'un module ou d'une bibliothèque. Ces différences affectent le comportement d'exécution des modules. Pour plus d'informations sur le mode d'analyse le plus efficace pour l'application, voir "Avantages du mode d'analyse lent par rapport au mode d'analyse syntaxique systématique.
- Un module peut être configuré pour être exécuté dans un des modes d'analyse syntaxique. Les bibliothèques peuvent être configurées pour prendre en charge l'un ou l'autre mode ou les deux modes. Une bibliothèque configurée pour prendre en charge les deux modes peuvent être référencée par un module utilisant le mode d'analyse syntaxique systématique et un module utilisant le mode d'analyse syntaxique lente. Le mode d'analyse syntaxique d'une bibliothèque lors de l'exécution est déterminé par les modules qui font référence à la bibliothèque. Lors de l'exécution, un module déclare son mode d'analyse syntaxique qui est utilisé par le module et les bibliothèques utilisés par le module.
- Les modules et les bibliothèques configurés pour différents modes d'analyse syntaxique sont compatibles dans les cas suivants :
  - Les modules et les bibliothèques configurés avec le mode d'analyse syntaxique lente sont compatibles avec les bibliothèques qui utilisent le mode d'analyse syntaxique lente ou les modes d'analyse syntaxique systématique et lente.
  - Les modules et les bibliothèques configurés avec le mode d'analyse syntaxique systématique sont compatibles avec les bibliothèques qui utilisent le mode d'analyse syntaxique systématique ou les modes d'analyse syntaxique systématique et lente.
  - Les bibliothèques configurées avec les modes d'analyse syntaxique systématique et lente sont compatibles uniquement avec les bibliothèques qui utilisent les modes d'analyse syntaxique systématique et lente.

- Utilisez le même mode d'analyse pour les modules qui interagissent qui communiquent en utilisant la liaison SCA. Si les modules communiquent en utilisant des modes d'analyse syntaxique différents, des problèmes de performance peuvent apparaître.

Concepts associés:

«Avantages comparés des deux modes d'analyse syntaxique : mode rapide et mode lent», à la page 158
Certaines applications tirent avantage du mode passif d'analyse syntaxique du XML alors que d'autres voient leurs performances améliorées avec le mode attentif d'analyse. Il est fortement conseillé de tester l'application dans les deux modes afin de déterminer lequel convient le mieux aux caractéristiques spécifiques de l'application.

Avantages comparés des deux modes d'analyse syntaxique : mode rapide et mode lent

Certaines applications tirent avantage du mode passif d'analyse syntaxique du XML alors que d'autres voient leurs performances améliorées avec le mode attentif d'analyse. Il est fortement conseillé de tester l'application dans les deux modes afin de déterminer lequel convient le mieux aux caractéristiques spécifiques de l'application.

Les applications qui analysent des flux importants de données XML constateront très probablement une amélioration de leurs performances si l'on utilise le mode lent. Ces améliorations se font d'autant plus sentir que la taille des flux d'octets XML augmente, alors que la quantité des données analysées par l'application dans ce flux diminue.

Les applications suivantes sont censées optimiser leurs performances dans ce mode :

- les applications qui analysent des flux de données non XML
- les applications utilisant des messages créés à l'aide du service BOFactory
- les applications qui analysent des messages XML de taille très modeste

Référence associée:

«Remarques relatives au choix du mode d'analyse syntaxique des objets métier», à la page 157
Le mode d'analyse syntaxique des objets métier détermine le mode d'analyse syntaxique des données XML. Un mode d'analyse syntaxique d'objet métier est défini dans un module ou une bibliothèque lors de sa création. Vous pouvez changer le mode d'analyse syntaxique du module ou de la bibliothèque, mais vous devez connaître les conséquences.

Remarques concernant la migration et le développement d'applications

Si vous configurez une application développée à l'origine avec le mode d'analyse syntaxique rapide et voulez utiliser le mode lent, ou si vous prévoyez d'utiliser alternativement les deux modes sur une application donnée, vous devez tenir compte des différences existant entre ces deux modes et de ce qu'implique le passage d'un mode à l'autre.

Gestion des erreurs

Des exceptions d'analyse syntaxique se produisent si le flux d'octets XML est malformé.

- En mode d'analyse syntaxique XML rapide, ces exceptions apparaissent dès que l'objet métier est analysé dans le flux XML entrant.
- Si c'est le mode lent qui est configuré, les exceptions se produisent tardivement au moment non seulement de l'accès aux propriétés de l'objet métier, mais encore à celui où est analysée la portion malformée du XML.

Pour traiter le XML malformé, vous avez plusieurs possibilités :

- déployer aux extrémités un bus de service d'entreprise pour valider le XML entrant
- concevoir une logique de détection passive des erreurs au niveau du point où l'accès se fait aux propriétés des objets métier

Piles et messages d'exceptions

L'implémentation sous-jacente des deux modes étant différente, les traces des piles générées par les interfaces de programmation et les services des objets métier ont beau porter le même nom de classe d'exception, elles risquent de contenir des messages différents pour les exceptions ou d'encapsuler de manière différente les classes d'exceptions.

Format de sérialisation XML

Le mode lent fournit une optimisation des performances qui, lors de la sérialisation, tente de copier vers le flux d'octets sortant le XML non modifié à partir du flux d'octets entrant. Il en résulte indéniablement un surcroît de performances, mais le format de la sérialisation du flux d'octets XML sortant risque d'être différent si la totalité de l'objet métier a été actualisé en mode lent ou s'il s'exécutait en mode rapide.

Bien que le format de la sérialisation du XML risque de ne pas être précisément équivalent du point de vue de la syntaxe, la valeur sémantique apportée par l'objet métier, elle, est équivalente quel que soit le mode d'analyse, et le XML peut être passé en toute sécurité entre des applications s'exécutant dans des modes d'analyse syntaxique équivalents sur le plan sémantique.

Validateur d'instances d'objets métier

Le validateur d'instances d'objets métier en mode d'analyse XML lente fournit une validation plus fidèle des objets métier, particulièrement la validation des facettes des valeurs des propriétés. Du fait de ces plus, le validateur d'instances en mode lent intercepte des problèmes que ne voit pas le mode rapide et il fournit des messages d'erreur plus détaillés.

XML Maps version 602

Les flux de médiation originellement développés avant la version 6.1 de WebSphere Integration Developer risquent de contenir des primitives Mappage qui utilisent une mappe ou une feuille de styles ne pouvant pas s'exécuter directement en mode d'analyse syntaxique XML lent. Lorsqu'on fait migrer une application pour l'utiliser en mode passif, les fichiers de mappes associés aux primitives Mappage peuvent être automatiquement mis à jour. Par contre, si une primitive Mappage se réfère directement à une feuille de styles qui a été éditée manuellement, cette feuille de styles ne migre pas et elle ne peut pas s'exécuter en mode d'analyse syntaxique XML lent.

API privées non publiées

Si une application exploite des interfaces de programmation d'objets métier spécifiques à une implémentation, la compilation de l'application a toutes les chances d'échouer lorsqu'on change de mode d'analyse syntaxique. En mode rapide, ces interfaces privées sont normalement des classes d'implémentation d'objets métier qui sont définies par Eclipse Modeling Framework (EMF).

Dans tous les cas, il est fortement recommandé que les API privées soit retirées de l'application.

API EMF d'objets de message de service

Un composant de médiation de IBM Integration Designer permet de manipuler le contenu des message à l'aide des classes et des interfaces Java fournies dans le package com.ibm.websphere.sibx.smobo. En mode lent, les interfaces Java de ce package sont toujours utilisables mais les méthodes qui se réfèrent directement aux classes et interfaces EMF ou qui sont hérités des interfaces EMF ont toutes les chances d'échouer.

En mode lent, il est impossible de transtyper en objets EMF ServiceMessageObject et son contenu.

Service BOMode

Le service BOMode sert à déterminer quel est le mode d'analyse du XML en cours d'exécution : mode rapide ou mode lent.

Migration

Toutes les applications antérieures à la version 7.0.0.0 s'exécutent en mode rapide. Lorsque, au moment de l'exécution, on les fait migrer à l'aide des outils de migration de l'environnement d'exécution BPM, elles continuent à s'exécuter en mode rapide.

Pour permettre à une application antérieure à la version 7.0.0.0 d'utiliser le mode lent, vous devez préalablement utiliser Integration Designer pour faire migrer les artefacts de l'application. Après la migration, vous pouvez alors configurer l'application pour qu'elle utilise le mode lent.

Voir Migration d'artefacts source pour savoir comment faire migrer des artefacts dans Integration Designer ; voir également Configuration du mode d'analyse syntaxique des objets métier des modules et des bibliothèques pour savoir comment définir le mode d'analyse.

Relations

Une relation est une association entre deux ou plusieurs entités de données, généralement des objets métier. Dans IBM Business Process Manager Advanced, les relations permettent de transformer des données équivalentes contenues dans plusieurs objets métier et d'autres données représentées différemment. Elles peuvent également être utilisées pour créer des associations entre différents objets métier se trouvant dans plusieurs applications. Il est possible de les partager sur différents produits, applications et solutions.

Le service de relation de IBM Business Process Manager Advanced fournit l'infrastructure et les opérations permettant de gérer les relations. Grâce à ses capacités de gestion des objets métier où qu'ils se trouvent, ce service peut fournir une vue unifiée et globale sur toutes les applications d'une entreprise et servir de bloc fonctionnel pour les solutions BPM. Extensibles et gérables, les relations peuvent être utilisées dans des solutions d'intégration complexes.

Définition d'une relation

Une relation est une association entre plusieurs objets métier. Chaque objet métier faisant partie d'une relation est appelé *participant* de cette relation. Chaque participant se distingue des autres par sa fonction (ou *rôle*) au sein de cette relation. Une relation contient une liste de rôles.

Sa *définition* détaille chaque rôle et indique la façon dont ils sont reliés. Elle détaille également la "forme" globale de la relation. Par exemple, un rôle donné peut n'avoir qu'un seul participant alors qu'un autre peut en disposer de plusieurs. Par exemple, vous pouvez définir une relation *voiture-propriétaire*, dans laquelle un propriétaire peut posséder plusieurs voitures. Autre exemple, une instance peut avoir plusieurs participants pour chacun des rôles suivants :

- Voiture (Ferrari)
- Propriétaire (Jean-Marc)

La définition de la relation constitue un modèle pour l'*instance* de relation. L'instance constitue l'instanciation de l'exécution de la relation. Dans l'exemple *voiture-propriétaire*, elle peut décrire n'importe laquelle des associations suivantes :

- Jean-Marc possède une Ferrari
- Sarah possède une Mazda
- Robert possède une Ferrari

Les relations vous permettent d'éviter d'avoir à personnaliser la persistance de suivi des relations dans votre logique métier. Dans certains cas, le service de relation fait tout le travail à votre place (voir l'exemple de la section concernant les relations d'identité).

Scénarios

Voici un exemple classique de situation dans laquelle une solution d'intégration peut avoir recours aux relations. Un groupe important achète plusieurs entreprises (ou unités métier). Chacune utilise différents logiciels pour effectuer un suivi du personnel et des ordinateurs portables. L'entreprise cherche à surveiller ses employés et ses ordinateurs portables et recherche une solution lui permettant de :

- Visualiser tous les employés des différentes unités métier comme s'ils étaient regroupés dans une base de données unique
- Avoir une vue unique de tous ses ordinateurs portables
- Permettre à ses employés de se connecter au système et d'acheter un ordinateur portable
- Intégrer les différents systèmes d'applications d'entreprise au niveau des unités métier

Pour cela, le groupe doit trouver un moyen de s'assurer, par exemple, que Jean-Marc Dupond et Jean-M. Dupond, qui figurent dans différentes applications, soient vus comme un seul et même employé. Par exemple, le groupe recherche un moyen de consolider une entité unique sur plusieurs espaces d'applications.

Les scénarios de relations plus complexes impliquent la mise en place de processus BPEL qui créent des relations entre différents objets se trouvant sur plusieurs applications. Grâce à de tels scénarios, les objets métier sont regroupés dans la solution d'intégration et ne sont plus disséminés dans les applications. Le service de relation fournit une plateforme permettant de gérer les relations de manière persistante (auparavant, vous deviez créer votre propre service de persistance d'objet). Les deux exemples suivants constituent des scénarios de relation complexes :

- Un objet métier **voiture** disposant d'un numéro VIN se trouve dans une application SAP et vous souhaitez effectuer un suivi du fait que cette voiture appartient à quelqu'un d'autre. Toutefois, la relation de propriété concerne une personne référencée dans une application PeopleSoft. Dans ce cas, deux solutions sont impliquées et vous devez créer une passerelle pour les relier.
- Un important groupe de distribution veut pouvoir effectuer un suivi des marchandises renvoyées contre un avoir ou un remboursement. Deux applications différentes sont impliquées : un système de gestion des commandes (pour les achats) et un système de gestion des retours (pour les retours). Les objets métier se trouvent dans plusieurs applications et vous devez trouver un moyen pour afficher les relations entre eux.

Modèles d'utilisation courants

Les modèles d'utilisation les plus courants sont les modèles d'*équivalence*, qui fonctionnent sur une base de références croisées ou de corrélations. Deux types de relations correspondent à ce modèle : *non-identité* et *identité*.

- Les **relations de non-identité** établissent des associations entre des objets métier ou d'autres données sur une base un à plusieurs ou plusieurs à plusieurs. Dans chaque instance de relation, il peut y avoir une ou plusieurs instances de chaque participant. La relation de recherche statique est un type de relation de non-identité. Exemple : une relation entre l'objet **CA** d'une application SAP et l'élément **Californie** d'une application Siebel.
- Les **relations d'identité** établissent des associations entre les objets métier ou d'autres données sur une base un à un. Dans chaque instance de relation, il ne peut y avoir qu'une seule instance de chaque participant. Les relations d'identité capturent les références croisées entre les objets métier qui sont équivalents au niveau sémantique mais qui sont identifiés différemment sur plusieurs applications. Chaque participant de la relation est associé à un objet métier disposant d'une valeur (ou d'une

combinaison de valeurs) qui identifie l'objet de façon unique. Les relations d'identité transforment généralement les attributs clés des objets métier, comme les ID numériques et les codes produit.

Par exemple, si vous disposez d'objets métier **voiture** dans des applications SAP, PeopleSoft et Siebel et si vous recherchez une solution permettant de les synchroniser, vous devez intégrer manuellement une logique de synchronisation des relations en six mappes :

SAP -> générique

générique -> SAP

PeopleSoft-> générique

générique -> PeopleSoft

Siebel-> générique

générique -> Siebel

Mais, si vous utilisez des relations dans votre solution, le service de relation fournit des implémentations prédéfinies de modèles qui gèrent à votre place toutes ces instances de relations.

Outils de gestion des relations

L'*éditeur de relations* contenu dans Integration Designer est un outil permettant de concevoir les rôles et les relations d'intégration métier. Pour obtenir des informations détaillées et des informations sur la création de relations et l'utilisation de l'éditeur de relations, reportez-vous à la rubrique Création de relations.

Le *service de relation* est un service d'infrastructure de IBM Business Process Manager qui administre les relations et les rôles du système et exécute les opérations de gestion des rôles et des relations.

Le *gestionnaire de relations* constitue l'interface d'administration permettant la gestion des relations. Pour y accéder, utilisez les pages correspondantes de la console d'administration.

Il est possible d'appeler des relations automatiquement à l'aide des API de service de relation.

Service de relation

Ce service stocke les données de relation dans les tables de relation où il effectue un suivi des valeurs spécifiques à l'application sur plusieurs applications et solutions. Il permet des opérations de gestion des relations et des rôles.

Fonctionnement de la relation

Les relations et les rôles sont définis à l'aide de l'interface graphique de l'éditeur de relations de Integration Designer. Le service de relation stocke les données de corrélation dans les tables de la base de données de relation, dans la source de données par défaut que vous définissez lors de la configuration du service de relation. Une autre table (parfois appelée "table de participant") stocke les informations de chaque participant de la relation. Le service de relation utilise ces tables pour effectuer un suivi des valeurs spécifiques à l'application correspondante et pour propager les informations mises à jour sur l'intégralité des solutions.

Les relations, qui sont des artefacts métier, sont déployées sur un projet ou une bibliothèque partagée. Au premier déploiement, le service de relation intègre les données.

Au moment de l'exécution, lorsque les mappes ou autres composants IBM Business Process Manager requièrent une instance de relation, les instances de la relation sont extraites ou mises à jour, selon le scénario choisi.

Les données d'instance de relation et de rôle peuvent être manipulées de trois façons :

- Appels des API du service de relation par des snippets Java du composant IBM Business Process Manager

- Transformations de relations dans le service de mappage d'objets métier de IBM Business Process Manager
- Utilisation de l'outil de gestion de relations

Pour obtenir des informations détaillées et des informations sur la création de relations et l'utilisation de l'éditeur de relations, l'identification des types de relations et l'utilisation de l'éditeur de relations, reportez-vous à la rubrique Création de relations.

Gestionnaire de relations

Le gestionnaire de relations constitue l'interface d'administration permettant la gestion des relations. Pour y accéder, utilisez les pages correspondantes de la console d'administration.

Le gestionnaire de relations offre une interface utilisateur graphique permettant de créer et de manipuler les données de rôles et de relations au moment de l'exécution. Vous pouvez gérer les entités de relation à tous les niveaux : instance de relation, instance de rôle, données d'attribut et données de propriété. Le gestionnaire de relations vous permet :

- D'afficher la liste des relations du système ainsi que des informations détaillées concernant les relations individuelles
- De gérer les instances de relation :
  - D'interroger les données de relation pour afficher les sous-ensembles de données d'instance
  - D'interroger les données de relation pour afficher les sous-ensembles de données d'instance à l'aide des vues de base de données
  - D'afficher la liste des instances de relation correspondant à une requête de relation ainsi que des informations détaillées concernant une instance
  - De modifier les valeurs de propriété d'une instance de relation
  - De créer et de supprimer des instances de relation
- De gérer les rôles et les instances de rôle :
  - D'afficher des informations détaillées concernant un rôle ou une instance de rôle
  - De modifier les propriétés d'une instance de rôle
  - De créer et de supprimer des instances de rôle correspondant à une relation
  - De restaurer des données d'instance de relation à un moment donné quand vous êtes sûr de la fiabilité de ces données
- D'importer des données d'une relation statique existante dans votre système ou d'exporter ces données vers un fichier RI ou CSV.
- De supprimer le modèle et les donnés d'une relation à partir du référentiel lors de la désinstallation de l'application qui l'utilise

Relations en environnement de déploiement réseau

Il est possible d'utiliser les environnements de déploiement réseau (ND) sans configuration supplémentaire.

En environnement de déploiement réseau (ND), les relations sont installées dans un cluster d'applications. Elles sont visibles dans ce cluster et tous les serveurs qui s'y trouvent ont alors accès aux données d'instance stockées dans la base de données de relations. La possibilité d'exécuter le service de relation dans un environnement ND rend ce dernier évolutif et hautement disponible.

Le gestionnaire de relations permet la gestion de relations sur plusieurs clusters via une interface d'administration centralisée. Pour connecter le gestionnaire de relations à un serveur faisant partie d'un cluster, sélectionnez sa relation MBean.

API de service de relation

Il est possible d'appeler des relations automatiquement à l'aide des API de service de relation, à partir ou hors des mappes d'objets métier.

Trois types d'API sont disponibles :

- API de manipulation d'instance de relation (dont création, mise à jour et suppression directe de données d'instance)
- API de prise en charge de patterns de relation (dont correlate(), correlateforeignKeyLookup)
- patterns de recherche de relation (API de recherche)

Bus de service d'entreprise de IBM Business Process Manager

IBM Business Process Manager prend en charge l'intégration des services d'application avec les mêmes fonctions que WebSphere Enterprise Service Bus.

Connexion de services via un bus de service d'entreprise

Avec un bus de service d'entreprise (ESB), vous pouvez optimiser la souplesse d'une architecture SOA. Les participants d'une interaction de service sont connectés à l'ESB, plutôt que directement à un autre module.

Quand le demandeur de services se connecte à l'ESB, l'ESB est responsable de la transmission de ses demandes, à l'aide de messages, au fournisseur de services proposant la fonction et la qualité de service requises. L'ESB simplifie les interactions demandeur-fournisseur et s'occupe de la non concordance des protocoles, des patterns d'interaction ou des fonctions de service. Un ESB peut également activer ou améliorer le contrôle et la gestion. L'ESB offre des fonctions de gestion et de virtualisation qui implémentent et étendent les principales fonctionnalités de l'architecture SOA.

L'ESB extrait les fonctions suivantes :

Emplacement et identité
: Les participants n'ont pas besoin de connaître l'emplacement ou l'identité des autres participants. Par exemple, les demandeurs n'ont pas besoin de savoir qu'une demande peut être traitée par n'importe lequel des nombreux fournisseurs ; les fournisseurs de services peuvent être ajoutés ou supprimés sans perturbation.

Protocole d'interaction
: Les participants n'ont pas besoin de partager le même protocole de communication ou le même style d'interaction. Par exemple, une demande exprimée en tant que SOAP via HTTP peut être gérée par un fournisseur comprenant uniquement SOAP via Java Message Service (JMS).

Interface
: Les demandeurs et les fournisseurs n'ont pas besoin de s'entendre sur une interface commune. Un ESB synchronise les différences en convertissant les messages de demande et de réponse dans un format attendu par le fournisseur.

Qualités de (interaction) service
: Les participants, ou administrateurs système, expriment leurs exigences en termes de qualité de service, notamment l'autorisation des demandes, le chiffrement et déchiffrement du contenu des messages, l'audit automatique des interactions de service, ainsi que l'acheminement souhaité de leur demandes (privilégiant la rapidité ou le coût, par exemple).

L'interposition de l'ESB entre les participants vous permet de moduler leurs interactions via une construction logique appelée *médiation*. Les médiations agissent sur les messages en cours entre les demandeurs et le fournisseurs. Par exemple, les médiations permettent de trouver des services avec des

caractéristiques spécifiques recherchées par un demandeur, ou de résoudre des différences d'interface entre demandeurs et fournisseurs. Pour les interactions complexes, les médiations peuvent être reliées successivement.

A l'aide des médiations, un bus de service d'entreprise exécute les actions suivantes entre le demandeur et le service :

- *Acheminement* des messages entre les services. Un bus de service d'entreprise offre une infrastructure de communication commune permettant de se connecter aux services et donc aux fonctions métier qu'ils représentent, sans que les programmeurs aient à écrire et gérer une logique de connectivité complexe.
- *Conversion* des protocoles de transport entre le demandeur et le service. Un bus de service d'entreprise est un moyen cohérent normalisé d'intégrer des fonctions métier qui utilisent des normes informatiques différentes. Cela permet de disposer de fonctions métier qui ne pourraient normalement pas communiquer, telles que la connexion d'applications dans des silos départementaux ou la participation des applications de différentes sociétés aux interactions de service.
- *Conversion* des formats de message entre le demandeur et le service. Un bus de service d'entreprise permet aux fonctions métier de d'échanger des informations dans des formats différents, le bus garantissant que l'information distribuée à une fonction métier a le format correspondant à l'application.
- *Traitement* des événements métier provenant de sources différentes. Un bus de service d'entreprise prend en charge les interactions basées sur l'événement en plus des échanges de message pour traiter les demandes de service.

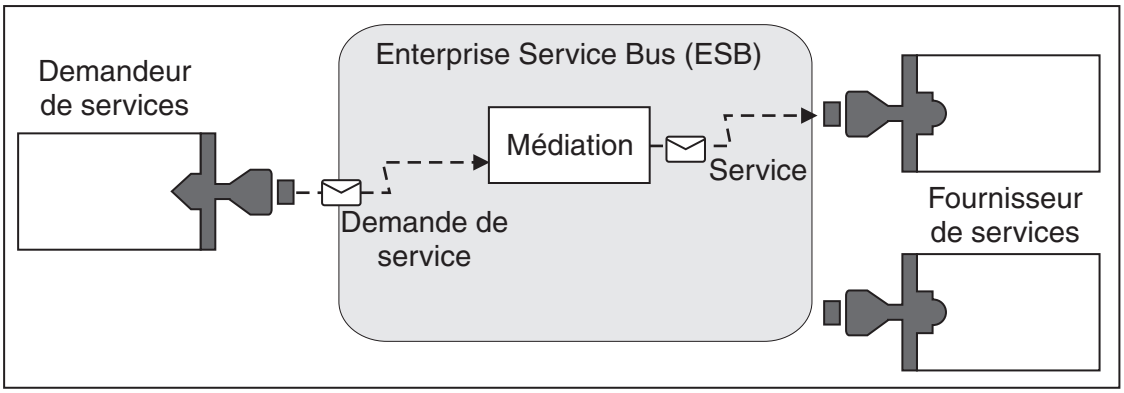


Figure 87. Bus de service d'entreprise. Le bus de service d'entreprise achemine les messages entre les applications, qui sont demandeurs ou fournisseurs de services. Le bus convertit les protocoles de transport ainsi que les formats des messages entre les demandeurs et les fournisseurs. Dans ce schéma, chaque application utilise un protocole différent (représenté par les différentes formes géométriques de leurs connecteurs) et utilise différents formats de message.

En utilisant le bus de service d'entreprise, vous vous consacrez désormais entièrement à votre coeur de métier, sans vous soucier des systèmes informatiques. Vous pouvez apporter des modifications ou des ajouts aux services quand vous avez besoin, par exemple, pour répondre aux évolutions de vos besoins métier, augmenter les capacités de service ou ajouter de nouvelles fonctionnalités. Vous pouvez effectuer vos modifications en redéfinissant le bus, avec très peu ou pas d'incidence sur les services et les applications existantes qui utilisent le bus.

Infrastructure de messagerie Enterprise Service Bus

IBM Business Process Manager inclut des fonctions de bus de services d'entreprise. IBM Business Process Manager prend en charge l'intégration de technologies orientées services, orientées messages et gérées par des événements pour fournir une infrastructure de messagerie normalisée au sein d'un bus de services d'entreprise intégré.

Les fonctions de services d'entreprise qui vous pouvez utiliser pour les applications d'entreprise fournissent non seulement une couche de transport mais également un support de médiation pour faciliter les interactions des services. Le bus de services d'entreprise s'appuie sur des normes ouvertes et l'architecture SOA (Service Oriented Architecture). Il repose sur l'infrastructure robuste Java EE et les services de plateformes associés fournis par IBM WebSphere Application Server Network Deployment.

IBM Business Process Manager est doté de la même technologie qu'IBM WebSphere Enterprise Service Bus. Cette technologie fait partie des fonctionnalités sous-jacentes de IBM Business Process Manager et aucune licence supplémentaire de WebSphere Enterprise Service n'est nécessaire pour en tirer parti.

Toutefois, vous pouvez déployer des licences autonomes supplémentaires de WebSphere Enterprise Service Bus dans l'entreprise pour étendre la connectivité des solutions d'intégration de processus IBM Business Process Manager. Par exemple, WebSphere Enterprise Service Bus peut être installé à proximité d'une application SAP pour héberger IBM WebSphere Adapter for SAP et transformer des messages avant d'envoyer des informations sur le réseau à un processus métier organisé par IBM Business Process Manager.

Hôtes de messagerie ou de destination de file d'attente

Un hôte de messagerie ou de destination de file d'attente constitue la fonction de messagerie au sein d'un serveur. Un serveur devient l'hôte de destination des messages lorsque vous le configurez en tant que cible de messagerie.

Le moteur de messagerie s'exécute dans le serveur. Le moteur de messagerie assure des fonctions de messagerie et constitue un point de connexion entre les applications et le bus. La communication asynchrone de l'architecture SCA (Service Component Architecture), les importations et les exportations JMS et le traitement interne asynchrone utilisent les files d'attente de messages sur le moteur de messagerie.

L'environnement de déploiement connecte la source de messages à la cible des messages via le bus, lorsque les modules d'applications sont déployés. Si vous connaissez la source et la cible des messages, vous pouvez déterminer plus facilement le type d'environnement de déploiement dont vous avez besoin.

Les données rémanentes peuvent être stockées dans un magasin de données par les applications. Un magasin de données est un ensemble de tables contenu dans une base de données ou dans un schéma, ou encore dans un magasin de données. Le moteur de messagerie utilise une instance d'une source de données JDBC pour interagir avec cette base de données.

Configurez l'hôte de destination des messages lorsque vous définissez votre environnement de déploiement **Server** à partir de la console d'administration, ou désignez le serveur en tant qu'hôte cible durant l'installation de logiciels.

Magasins de données :

Chaque moteur de messagerie peut utiliser un magasin de données (un magasin de données constitue un ensemble de tables contenues dans une base de données ou dans un schéma incluant des données permanentes).

Toutes les tables du magasin sont contenues dans le même schéma de base de données. Vous pouvez créer chaque magasin de données dans une base de données séparée. Il est également possible de créer plusieurs magasins dans la même base, chaque magasin utilisant un schéma différent.

Un moteur de messagerie utilise une instance d'une source de données JDBC pour interagir avec la base de données qui contient le magasin de données pour ce moteur de messagerie.

Fournisseurs JDBC
Vous pouvez utiliser les fournisseurs JDBC pour que les applications puissent interagir avec les bases de données relationnelles.

Les applications utilisent des fournisseurs JDBC pour interagir avec des bases de données relationnelles. Le fournisseur JDBC fournit la classe d'implémentation de pilote JDBC qui est nécessaire pour accéder à une base de données spécifique. Pour créer un pool de connexions à cette base de données, associez une source de données au fournisseur JDBC. Ensemble, le fournisseur JDBC et les objets de source de données présentent les mêmes fonctions que la fabrique de connexions Java Connector Architecture (JCA), qui assure la connexion à une base de données non relationnelle.

Voir les exemples de configuration d'environnement autonome type et de configuration d'environnement de déploiement type dans la rubrique précédente.

Pour plus d'informations sur les fournisseurs JDBC, voir «Fournisseurs JDBC» dans le centre de documentation de WebSphere Application Server.

Bus d'intégration de services pour IBM Business Process Manager
Un bus d'intégration de services est un mécanisme de communications géré prenant en charge l'intégration de services via une messagerie synchrone et asynchrone. Un bus se compose de moteurs de messagerie interconnectés gérant les ressources de bus. Il représente l'une des technologies WebSphere Application Server sur lesquelles repose IBM Business Process Manager.

Un bus d'intégration de services unique et moteur de messagerie unique utilise le même schéma de base de données que la base de données produit par défaut. Chaque environnement de déploiement possède son propre bus. Le bus unique est appelé **BPM.*nom_environnement_déploiement*.Bus**.

Une destination de bus est une adresse logique vers laquelle les applications peuvent définir une liaison en tant que fournisseur, consommateur, ou les deux. Une destination de file d'attente est une destination de bus utilisée pour la messagerie point-à-point.

Chaque bus peut contenir un ou plusieurs membres, chacun d'eux étant soit un serveur, soit un cluster.

Le terme de *topologie de bus* se rapporte à l'organisation physique entre les serveurs d'applications, les moteurs de messagerie et les gestionnaires de files d'attente WebSphere MQ, ainsi que le modèle de connexions et de liaisons de bus intermédiaires qui composent le bus ESB.

Modules et applications de service
Un module de service est un module SCA (Service Component Architecture) qui offre des services dans l'environnement d'exécution. Lorsque vous déployez un module de service sur IBM Business Process Manager, vous générez l'application de service associée conditionnée sous la forme d'un fichier EAR (enterprise archive).

Les modules de service sont les unités de base d'un déploiement et peuvent contenir des composants, des bibliothèques et des modules de transfert dont se sert l'application de service associée. Les modules de service disposent d'exportation et à titre facultatif d'importations pour définir les relations entre modules et demandeurs et fournisseurs de services. WebSphere Process Server prend en charge les modules pour les services métier et les modules de médiation. Les modules et les modules de médiation constituent des types de modules SCA. Un module de communication permet les communications entre applications en transformant l'appel de service dans un format compris par la cible, en transmettant la demande à la cible et en renvoyant le résultat au module émetteur. Un module pour un service métier met en oeuvre la logique d'un processus métier. Toutefois, un module peut aussi inclure la même logique de médiation que celle conditionnée dans le module de médiation.

Déploiement d'une application de service

Le processus de déploiement d'un fichier EAR contenant une application de service est identique à celui de tout fichier EAR. Vous pouvez modifier les valeurs des paramètres de médiation lors de la phase d'exécution. Après avoir déployé un fichier EAR contenant un module SCA, vous pouvez visualiser les informations sur l'application de service et son module associé. Vous pouvez visualiser la manière dont un module de service est connecté au demandeurs de services (via les exportations) et aux fournisseurs de services (via les importations).

Affichage des détails d'un module SCA

Les informations de module de service que vous pouvez afficher dépendent du module SCA. Elles comprennent les attributs suivants.

- Nom du module SCA
- Description du module SCA
- Nom de l'application associée
- Version du module SCA, si le module est versionné
- Importations du module SCA :
  - Les interfaces d'importation sont des définitions abstraites qui décrivent la façon dont un module SCA accède à un service.
  - Les liaisons d'importation sont des définitions concrètes qui indiquent le mécanisme physique via lequel un module SCA accède à un service. Par exemple, via SOAP/HTTP.
- Exportations du module SCA :
  - Les interfaces d'exportation sont des définitions abstraites qui décrivent la façon dont les demandeurs de services accèdent à un module SCA.
  - Les liaisons d'exportation sont des définitions concrètes qui indiquent le mécanisme physique via lequel un demandeur de services accède à un modules SCA et indirectement un service.
- Propriétés de module SCA

Importations et liaisons d'importation

Les importations définissent les interactions entre les modules SCA et les fournisseurs de services. Les modules SCA utilisent des importations pour permettre aux composants d'accéder aux services externes (services qui se trouvent en dehors du module SVA) à l'aide d'une représentation locale. Les liaisons d'importation définissent la façon spécifique dont on accède à un service externe.

Si les modules SCA n'ont pas besoin d'accéder à des services externes, ils n'ont pas besoin de disposer d'importations. Les modules de médiation disposent généralement d'une ou de plusieurs importations qui sont utilisées pour transmettre les messages ou demandes sur leurs cibles prévues.

Interfaces et liaisons

Une importation de modules SCA requiert au moins une interface et qu'une importation de modules SCAN ait une seule liaison.

- Les interfaces d'exportation sont des définitions abstraites qui définissent un ensemble d'opérations à l'aide de WSDL (Web Services Description Language), un langage XML permettant de décrire des services Web. Un module SCA peut avoir un grand nombre d'interfaces.
- Les liaisons d'importation sont des définitions concrètes qui spécifient le mécanisme physique utilisé par les modules SCA pour accéder à un service externe.

Liaisons d'importation prises en charge

IBM Business Process Manager prend en charge les liaisons d'importation ci-après :

- Les liaisons SCA connectent les modules SCA à d'autres modules SCA. Les liaisons SCA sont également appelées liaisons par défaut.
- Les liaisons de service Web permettent aux composants d'appeler des services Web. Les protocoles pris en charge sont SOAP1.1/HTTP, SOAP1.2/HTTP et SOAP1.1/JMS.

  Vous pouvez utiliser une liaison SOAP1.1/HTTP ou SOAP1.2/HTTP basée sur JAX-WS (Java API for XML Web Services), qui permet l'interaction avec les services à l'aide de liaisons de document ou de liaisons littérales RPC et qui utilise des gestionnaires JAX-WS pour personnaliser les appels. Une liaison SOAP1.1/HTTP distincte est fournie pour permettre l'interaction avec les services qui utilisent une liaison codée RPC ou lorsque des gestionnaires JAX-RPC doivent être utilisés pour personnaliser les appels.
- Les liaisons HTTP permettent d'accéder aux applications à l'aide du protocole HTTP.
- Les liaisons d'importation EJB (Enterprise JavaBeans)permettent aux composants SCA d'appeler des services fournis par la logique métier Java EE exécutée sur un serveur Java EE.
- Les liaisons de système d'information d'entreprise (EIS) permettent la connectivité entre les composants SCA et un système EIS externe. Cette communication est obtenue en utilisant des adaptateurs de ressources.
- Les liaisons Java Message Service (JMS) 1.1 permettent l'interopérabilité avec le fournisseur de messagerie par défaut de WebSphere Application Server. JMS peut exploiter divers types de protocoles de transport, tels que TCP/IP et HTTPS. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge.
- Les liaisons JMS génériques permettent l'interopérabilité avec les fournisseurs JMS tiers qui s'intègrent à WebSphere Application Server à l'aide de la fonction JMS Application Server Facility (ASF).
- Les liaisons JMS WebSphere MQ permettent l'interopérabilité avec les fournisseurs JMS basés sur WebSphere MQ. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge. Si vous souhaitez utiliser WebSphere MQ comme fournisseur JMS, utilisez des liaisons JMS WebSphere MQ.
- Les liaisons WebSphere MQ permettent l'interopérabilité avec WebSphere MQ. Vous ne pouvez utiliser des liaisons WebSphere MQ qu'avec des gestionnaires de file d'attente distants via une connexion client WebSphere MQ ; en effet, vous ne pouvez pas les utiliser avec des gestionnaires de file d'attente locaux. Utilisez des liaisons WebSphere MQ si vous souhaitez communiquer avec des applications natives WebSphere MQ.

Appel dynamique de services

Les services peuvent être appelés via n'importe quelle liaison d'importation prise en charge. Un service se trouve généralement sur un point de contact spécifié lors de l'importation. Ce point de contact est appelé point de contact statique. Il est possible d'appeler un service différent en remplaçant le point de contact statique. Le remplacement dynamique des points de contact statiques vous permet d'appeler un service sur un autre point de contact, via n'importe quelle liaison d'importation prise en charge. L'appel dynamique de services vous permet également d'appeler un service pour lequel la liaison d'importation prise en charge n'a pas de point de contact statique.

Une importation avec une liaison associée est utilisée pour spécifier le protocole et sa configuration pour l'appel dynamique. L'importation utilisée pour l'appel dynamique peut être reliée au composant appelant ou être sélectionnée dynamiquement lors de la phase d'exécution.

Pour les appels de service Web et SCA, il est également possible d'effectuer un appel dynamique sans importation, avec le protocole et la configuration extraite de l'URL du point de contact. Le type de cible de l'appel est identifié à partir de l'adresse URL du point de contact. Si une importation est utilisée, l'URL doit être compatible avec le protocole de la liaison d'importation.

- Une adresse URL SCA URL indique l'appel d'un autre module SCA.

- Une adresse URL HTTP ou JMS par défaut indique l'appel d'un service Web ; pour ces adresses URL, il est possible de fournir une valeur supplémentaire pour le type de liaison qui indique que l'adresse URL représente un appel via une liaison HTTP ou JMS.
- Pour une adresse URL HTTP de service Web, le protocole SOAP 1.1 est utilisé par défaut et une valeur pour le type de liaison indiquant l'utilisation de SOAP 1.2 peut être spécifiée.

Exportation et liaisons d'exportation

Les exportations définissent les interactions entre les modules SCA et les demandeurs de services. Les modules SCA utilisent des exportations pour proposer des services aux autres. Les liaisons d'exportation définissent le mode d'accès à un module SCA par les demandeurs de service.

Interfaces et liaisons

Une exportation de module SCA nécessite au moins une interface.

- Les interfaces d'exportation sont des définitions abstraites qui définissent un ensemble d'opérations à l'aide de WSDL (Web Services Description Language), un langage XML permettant de décrire des services Web. Un module SCA peut disposer d'un grand nombre d'interfaces d'exportation.
- Les liaisons d'exportation sont des définitions concrètes qui spécifient le mécanisme physique utilisé par les demandeurs de services pour accéder à un service. En règle générale, une exportation de module SCA a une liaison définie. Une exportation, pour laquelle aucune liaison n'a été spécifiée, est interprétée par l'environnement d'exportation comme une exportation dotée d'une liaison de type SCA.

Liaisons d'exportation prises en charge

Le IBM Business Process Manager prend en charge les liaisons d'exportation ci-après :

- Les liaisons SCA connectent les modules SCA à d'autres modules SCA. Les liaisons SCA sont également appelées liaisons par défaut.
- Les liaisons de services Web permettent d'appeler des exportations en tant que services Web. Les protocoles pris en charge sont SOAP1.1/HTTP, SOAP1.2/HTTP et SOAP1.1/JMS.

  Vous pouvez utiliser une liaison SOAP1.1/HTTP ou SOAP1.2/HTTP basée sur JAX-WS (Java API for XML Web Services), qui permet l'interaction avec les services à l'aide de liaisons de document ou de liaisons littérales RPC et qui utilise des gestionnaires JAX-WS pour personnaliser les appels. Une liaison SOAP1.1/HTTP distincte est fournie pour permettre l'interaction avec les services qui utilisent une liaison codée RPC ou lorsque des gestionnaires JAX-RPC doivent être utilisés pour personnaliser les appels.
- Les liaisons HTTP permettent d'accéder aux exportations à l'aide du protocole HTTP.
- Les liaisons d'exportation EJB (Enterprise JavaBeans) permettent d'exposer les composants SCA comme des EJB pour que la logique métier de Java EE puisse appeler les composants SCA qui ne seraient autrement pas disponibles.
- Les liaisons de système d'information d'entreprise (EIS) permettent la connectivité entre les composants SCA et un système EIS externe. Cette communication est obtenue en utilisant des adaptateurs de ressources.
- Les liaisons Java Message Service (JMS) 1.1 permettent l'interopérabilité avec le fournisseur de messagerie par défaut de WebSphere Application Server. JMS peut exploiter divers types de protocoles de transport, tels que TCP/IP et HTTPS. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge.
- Les liaisons JMS génériques permettent l'interopérabilité avec les fournisseurs JMS tiers qui s'intègrent à WebSphere Application Server à l'aide de la fonction JMS Application Server Facility (ASF).
- Les liaisons JMS WebSphere MQ permettent l'interopérabilité avec les fournisseurs JMS basés sur WebSphere MQ. La classe de message JMS et ses cinq sous-types (Text, Bytes, Object, Stream et Map) sont automatiquement pris en charge. Si vous souhaitez utiliser WebSphere MQ comme fournisseur JMS, utilisez des liaisons JMS WebSphere MQ.

- Les liaisons WebSphere MQ permettent l'interopérabilité avec WebSphere MQ. Utilisez une connexion éloignée (ou client) pour vous connecter à un gestionnaire de files d'attente MQ sur un poste distant. Une connexion (ou liaisons) locale correspond à une connexion directe à WebSphere MQ. Elles ne peuvent être utilisées que pour la connexion à un gestionnaire de files d'attente MQ sur une même machine. WebSphere MQ autorise les deux types de connexion, mais les liaisons MQ prennent en charge uniquement la connexion "à distance" (ou "client") .

Modules de médiation

Les modules de médiation sont des modules SCA qui peuvent modifier le format, le contenu ou la cible des demandes de services.

Les modules de médiation s'appliquent aux messages circulant entre les demandeurs et les fournisseurs de services. Vous pouvez acheminer des messages vers différents fournisseurs de services et de modifier le format ou le contenu du message. Les modules de médiation peuvent fournir des fonctions telles que la consignation des messages et le traitement des erreurs, personnalisées en fonction de votre environnement.

Vous pouvez modifier certains aspects des modules de médiation à partir de la console d'administration , sans que le redéploiement du module soit nécessaire.

Composants des modules de médiation

Les modules de médiation contiennent les éléments suivants :

- Importations qui définissent des interactions entre les modules SCA et les fournisseurs de services. Elles permettent aux modules SCA d'appeler des services externes comme s'ils étaient locaux. Vous pouvez visualiser les importations du module de médiation et modifier la liaison.
- Exportations qui définissent des interactions entre les modules SCA et les demandeurs de services. Elles permettent à un module SCA d'offrir un service et de définir les interfaces externes (points d'accès) d'un module SCA. Vous pouvez visualiser des exportations de module de médiation.
- Composants SCA, blocs structurels des modules SCA tels que les modules de médiation. Vous pouvez créer et personnaliser des composants et des modules SCA graphiquement via Integration Designer. Après avoir déployé un module de médiation, vous pouvez en personnaliser certains aspects à partir de la console d'administration , sans que le redéploiement du module soit nécessaire.

  Généralement, les modules de médiation contiennent un type spécifique de composant SCA appelé *composant de flux de médiation*. Les composants de flux de médiation permettent de définir ces flux.

  Un composant de flux de médiation peut contenir aucune, une ou plusieurs primitives de médiation. IBM Business Process Manager prend en charge un ensemble fourni de primitives de médiation qui fournissent des fonctionnalités pour l'acheminement et la transformation de messages. Si vous avez besoin d'une primitive de médiation plus souple, utilisez la primitive de médiation personnalisée pour appeler la logique personnalisée.

  L'objet d'un module de médiation qui ne contient pas de composant de flux de médiation est de convertir des demandes de services d'un protocole à un autre. Par exemple, une demande de service peut être effectuée via SOAP/JMS, mais risque d'avoir besoin d'être transformée en SOAP/HTTP avant d'être envoyée.

  Remarque : Vous pouvez afficher les modules de médiation et leur apporter certaines modifications à partir d'IBM Business Process Manager. Cependant, il n'est pas possible de visualiser ni de modifier des composants SCA à l'intérieur d'un module à partir d'IBM Business Process Manager. Utilisez Integration Designer pour personnaliser les composants SCA.

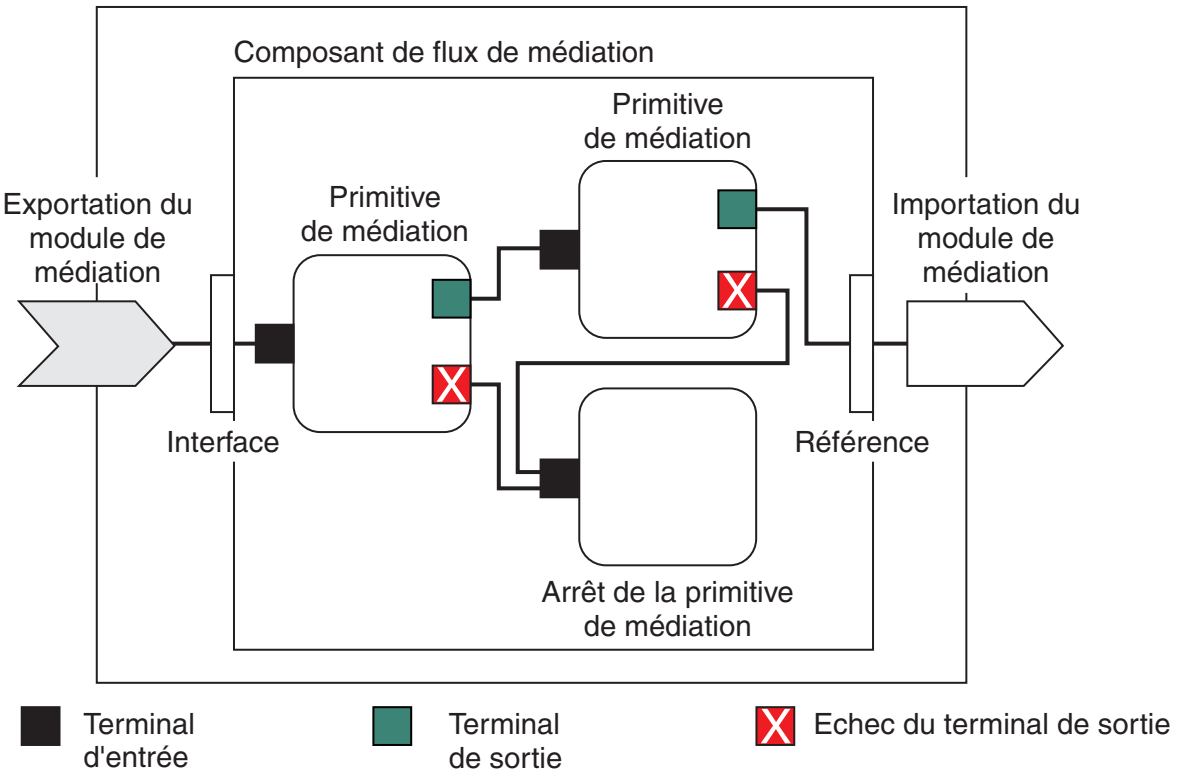


Figure 88. Exemple simplifié d'un module de médiation. Le module de médiation contient un composant de flux de médiation, qui contient des primitives de médiation.

- Propriétés

  Les propriétés de certaines primitives de médiation peuvent être affichées sur la console d'administration en tant que propriétés complémentaires d'un module SCA.

  Pour que les propriétés de la primitive de médiation soient visibles à partir de la console d'administration d'IBM Business Process Manager, le développeur d'intégration doit promouvoir les propriétés. Certaines propriétés peuvent être configurées administrativement et Integration Designer les décrit comme étant des propriétés pouvant être promues du cycle d'intégration au cycle d'administration. La raison pour laquelle d'autres propriétés sont incompatibles avec une configuration administrative est que leur modification affecte le flux de médiation d'une manière qui nécessite le redéploiement du module de médiation. Integration Designer répertorie les propriétés que vous pouvez promouvoir dans la liste des Propriétés promues d'une primitive de médiation.

  Vous pouvez utiliser la console d'administration d'IBM Business Process Manager pour modifier la valeur des propriétés promues sans qu'il soit nécessaire de redéployer un module de médiation ni de redémarrer le serveur ou le module.

  Généralement, les flux de médiation utilisent immédiatement les modifications de propriété. Toutefois, si les modifications de propriété se produisent dans une cellule de gestionnaire de déploiement, elles prendront effet sur chaque noeud à chaque fois qu'il sera synchronisé. Par ailleurs, les flux de médiation qui sont en transit continuent d'utiliser les valeurs précédentes.

  Remarque : A partir de la console d'administration, vous ne pouvez modifier que les valeurs de propriété et non pas les groupes, noms ou types de propriété. Si vous souhaitez modifier les groupes, noms ou types de propriété, vous devez utiliser Integration Designer.
- Un module de médiation ou une bibliothèque dépendante peut également définir des sous-flux. Un sous-flux encapsule un ensemble de primitives de médiation connectées les unes aux autres comme un élément réutilisable d'une logique d'intégration. Une primitive peut être ajoutée à un flux de médiation pour appeler un sous-flux.

Déploiement de modules de médiation

Les modules de médiation sont créés via Integration Designer et sont généralement déployés sur IBM Business Process Manager dans un fichier EAR (fichier d'archive d'entreprise).

La valeur des propriétés promues peut être modifiée lors du déploiement.

Vous pouvez exporter un module de médiation à partir de Integration Designer, puis ordonner à Integration Designer de compiler le module de médiation dans un fichier JAR (archive Java), lequel est ensuite intégré à un fichier EAR. Vous pouvez maintenant déployer le fichier EAR en installant une nouvelle application à partir de la console d'administration.

Les modules de médiation peuvent être considérés comme une entité. En revanche, les modules SCA sont définis par un certain nombre de fichiers XML stockés dans un fichier JAR.

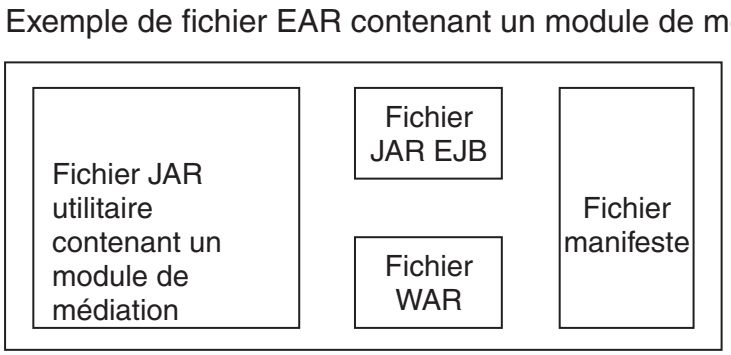


Figure 89. Exemple simplifié de fichier EAR contenant un module de médiation. Le fichier EAR contient des fichiers JAR. Le fichier JAR utilitaire contient un module de médiation.

Primitives de médiation

Les composants de flux de médiation agissent sur les flux de messages entre les composants de service. Les fonctionnalités d'un composant de médiation sont implémentées par les *primitives de médiation*, qui mettent en oeuvre des types d'implémentation de service standard.

Un composant de flux de médiation dispose d'un ou plusieurs flux. Par exemple, un pour la requête et un pour la réponse.

IBM Business Process Manager prend en charge un ensemble de primitives de médiation qui implémentent des fonctionnalités de médiation standard pour les modules de médiation ou modules déployés vers IBM Business Process Manager. Si vous avez besoin d'utiliser des fonctions de médiation spéciales, vous pouvez développer vos propres primitives de médiation.

Une primitive de médiation définit une opération «entrante» qui traite ou gère les messages représentés sous forme d'objets SMO (Service Message Object). Elle peut également définir une opération «sortante» qui envoie des messages vers un autre composant ou module.

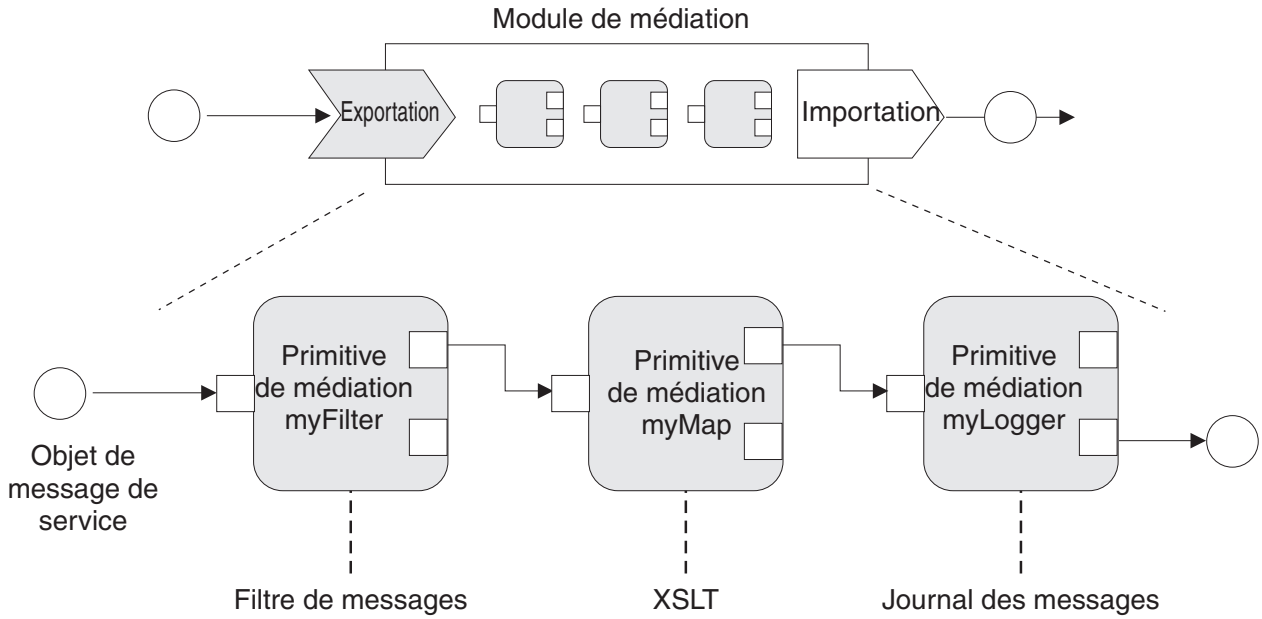


Figure 90. Module de médiation contenant trois primitives de médiation

Vous pouvez utiliser Integration Designer pour configurer les primitives de médiation et définir leurs propriétés. Certaines de ces propriétés peuvent être visibles pour l'administrateur d'exécution si elles ont été promues. Toute propriété primitive de médiation qui peut être promue peut également être une propriété dynamique. Une propriété dynamique peut être remplacée, en phase d'exécution, à l'aide d'un fichier de règles.

Integration Designer permet également de modéliser et d'assembler sous forme graphique les composants de flux de médiation à partir de primitives de médiation et d'assembler les modules de médiation ou les modules depuis des composants de flux de médiation. La console d'administration fait référence aux modules de médiation et modules en tant que modules SCA.

Integration Designer permet également la définition de sous-flux dans les modules ou leurs bibliothèques dépendantes. Un sous-flux peut contenir toute primitive de médiation excepté pour la primitive de médiation Résolution de règle. Un sous-flux est appelé à partir d'un flux de demandes ou de réponses ou à partir d'un autre sous-flux utilisant la primitive de médiation Sous-flux. Les propriétés promues à partir des primitives de médiation dans un sous-flux sont affichées en tant que propriétés sur les primitives de médiation Sous-flux. Elles peuvent ensuite être à nouveau promues jusqu'à ce qu'elles atteignent le niveau de module auquel elles peuvent être modifiées par l'administrateur d'exécution.

Primitives de médiation prises en charge

L'ensemble suivant de primitives de médiation est pris en charge par IBM Business Process Manager:

Mappe d'objet métier
Transforme les messages.

- Définit les transformations de message à l'aide d'une mappe d'objet métier, qui peut être réutilisée.
- Permet de définir les transformations de message sous forme graphique, à l'aide de l'éditeur de mappe d'objet métier.
- Peut modifier le contenu d'un message.
- Peut transformer un type de message d'entrée en un type de message de sortie différent.

Médiation personnalisée

Permet d'implémenter votre propre logique de médiation en code Java. La primitive de médiation personnalisée associe la flexibilité d'une primitive de médiation définie par l'utilisateur, à la simplicité d'une primitive de médiation prédéfinie. Vous pouvez créer des patterns d'acheminement et de transformation complexes en :

- Créant le code Java.
- Créant vos propres propriétés.
- Ajoutant de nouveaux terminaux.

Vous pouvez appeler un service depuis une primitive de médiation personnalisée, mais la primitive de médiation d'appel de service (Service Invoke) est conçue pour appeler des services et fournir d'autres fonctions, notamment de relance.

Gestionnaire de données

Vous permet de transformer une partie d'un message. Il est utilisé pour convertir un élément de message d'un format physique en une structure logique ou d'une structure logique en format physique. L'utilisation principale de la primitive consiste à convertir un format physique, comme une chaîne de texte au sein d'un objet de message texte JMS, en une structure d'objet métier logique et inversement. Cette médiation est généralement utilisée pour :

- Transformer une section du message entrant d'une structure définie en une autre, par exemple lorsque l'objet SMO comprend une valeur de chaîne délimitée par une virgule et que vous voulez faire une analyse syntaxique dans un objet métier spécifique.
- Modifier le type de message – par exemple lorsqu'une exportation JMS a été configurée pour utiliser une liaison de données de type de base JMS et qu'au sein du module de médiation, le développeur d'intégration décide que le contenu doit être gonflé en une structure BO.

Consultation de base de données

Modifie les messages, à l'aide d'informations provenant d'une base de données fournie par l'utilisateur.

- Vous devez définir une base de données, une source de données et tous les paramètres d'authentification du serveur que doit utiliser la primitive de médiation Recherche de base de données à utiliser. Utilisez la console d'administration pour vous simplifier la tâche.
- La primitive de médiation Recherche de base de données ne peut lire qu'à partir d'une seule table.
- La colonne de clé spécifiée doit contenir une valeur unique.
- Les données des colonnes de valeur doivent être d'un type de schéma XML simple ou d'un type de schéma XML permettant d'étendre un type de schéma XML simple.

Consultation de point de contact (Endpoint Lookup)

Permet d'effectuer le routage dynamique de requêtes en recherchant les points de contact de services dans un référentiel.

- Les informations relatives au point de contact de service sont extraites de WebSphere Service Registry and Repository (WSRR). Le registre WSRR peut être local ou distant.
- Vous effectuez les modifications du registre à partir de la console d'administration WSRR.
- IBM Business Process Manager doit connaître le registre à utiliser, par conséquent, vous devez créer des définitions d'accès WSRR à l'aide de la console d'administration IBM Business Process Manager.

Emetteur d'événements

Améliore le contrôle en vous laissant envoyer des événements à partir d'un composant de flux de médiation.

- Vous pouvez suspendre l'action de médiation en décochant la case.
- Vous pouvez visualiser ces événements via le navigateur CBE (Common Base Event) de IBM Business Process Manager.

- Vous pouvez uniquement envoyer des événements vers un point significatif d'un flux de médiation, à des fins de performances.
- Vous pouvez définir les parties du message que contient l'événement.
- Les événements sont envoyés suivant le format Common Base Events (CBE) vers un serveur Common Event Infrastructure (CEI).
- Pour pouvoir exploiter pleinement les informations sur les émetteurs d'événements, les consommateurs d'événements doivent comprendre parfaitement la structure Common Base Events. Le format Common Base Events est caractérisé par un schéma global, mais qui ne modélise pas les données spécifiques à l'application contenues dans les éléments de données étendus. Afin de modéliser les éléments de données étendus, les outils Integration Designer génèrent un fichier de définitions pour le catalogue d'événements Common Event Infrastructure (CEI), pour chacune des primitives de médiation Emetteur d'événement. Les fichiers de définitions du catalogue d'événements sont des artefacts d'exportation destinés à vous venir en aide ; ils ne sont pas utilisés par Integration Designer ou par le programme d'exécution IBM Business Process Manager. Il convient de vous référer aux fichiers de définitions du catalogue d'événements lorsque vous créez des applications destinées à consommer des événements générés par un émetteur d'événements.
- Vous pouvez spécifier d'autres options de contrôle à partir de IBM Business Process Manager. Ainsi, vous pouvez surveiller les événements émis à partir d'importations et d'exportations.

Fail (Echec)
Arrête un chemin donné dans le flux, et génère une exception.

Fan In (Entrance)
Permet de regrouper (d'associer) des messages.
- Est utilisable uniquement en association avec la primitive de médiation de sortance.
- L'association des primitives de médiation d'entrance et de sortance permet le regroupement de données dans un message de sortie.
- La primitive de médiation d'entrance reçoit des messages jusqu'à ce qu'un point de décision soit atteint, puis un message est sorti.
- Le contexte partagé permet de conserver les données de regroupement.

Fan Out (Sortance)
Permet de diviser et de regrouper (associer) des messages.
- L'association des primitives de médiation d'entrance et de sortance permet le regroupement de données dans un message de sortie.
- En mode d'itération, la primitive de médiation de sortance vous permet d'itérer via un seul message d'entrée contenant un élément répétitif. Pour chaque occurrence de l'élément répétitif, un message est envoyé.
- Le contexte partagé permet de conserver les données de regroupement.

Configurateur d'en-tête HTTP
Fournit un mécanisme de gestion des en-têtes dans les messages HTTP.
- Peut créer, définir, copier ou supprimer des en-têtes de messages HTTP.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes HTTP.

Mappage
Transforme les messages.
- Permet d'effectuer des transformations XSL (Extensible Stylesheet Language) ou des transformations de mappe d'objets métier.
- Vous transformez les messages avec la transformation XSLT 1.0, XSLT 2.0 ou de mappe d'objets métier. Les transformations XSL opèrent sur une sérialisation XML du message, tandis que les transformations de Mappe d'objet métier opèrent sur les objets SDO.

Configurateur d'élément de message

Fournit un système simple permettant de définir le contenu des messages.

- Permet de modifier, ajouter ou supprimer les éléments du message.
- Ne modifie pas le type du message.
- Les données des colonnes de valeur doivent être d'un type de schéma XML simple ou d'un type de schéma XML permettant d'étendre un type de schéma XML simple.

Filtre de message

Achemine les messages par différent chemins, selon le contenu du message.

- Vous pouvez suspendre l'action de médiation en décochant la case.

Journal des messages

Consigne les messages dans une base de données relationnelle ou via votre propre consignateur personnalisé. Les messages sont stockés au format XML, c'est pourquoi les données peuvent subir un post-traitement par des applications compatibles XML.

- Vous pouvez suspendre l'action de médiation en décochant la case.
- Le schéma de base de données rationnel (structure de table) est défini par IBM.
- Par défaut, la primitive de médiation Message Logger utilise la base de données Common. L'environnement d'exécution mappe la source de données de **jdbc/mediation/messageLog** dans la base de données Common.
- Vous pouvez définir les classes d'implémentation Handler pour personnaliser le comportement du consignateur personnalisé. Vous pouvez éventuellement fournir des classes d'implémentation Formatter, Filter ou les deux pour personnaliser le comportement du consignateur personnalisé.

Configurateur d'en-tête MQ

Fournit un mécanisme de gestion des en-têtes dans les messages MQ.

- Peut créer, définir, copier ou supprimer des en-têtes de messages MQ.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes MQ.

Résolution de règle

Permet la configuration dynamique des demandes en recherchant les points de contact de services et les fichiers de règles associés dans un référentiel.

- Vous pouvez utiliser un fichier de règles pour remplacer de manière dynamique les propriétés promues d'autres primitives de médiation.
- Les informations relatives au point de contact de service et les informations de règle sont extraites de WebSphere Service Registry and Repository (WSRR). Le registre WSRR peut être local ou distant.
- Vous effectuez les modifications du registre à partir de la console d'administration WSRR.
- IBM Business Process Manager doit connaître le registre à utiliser, par conséquent, vous devez créer des définitions d'accès WSRR à l'aide de la console d'administration IBM Business Process Manager.

Invocation de service

Appel un service depuis un flux de médiation, au lieu d'attendre jusqu'à la fin du flux de médiation et d'utiliser un système d'appel.

- Si le service renvoie une erreur, vous pouvez tenter de nouveau le même service ou appeler un autre service.
- La primitive de médiation d'appel de service (Service Invoke) est une primitive de médiation puissante qui peut être utilisée pour des appels de service simples, ou conjointement avec d'autres primitives de médiation pour les médiations complexes.

Définir un type de message

En phase de développement d'intégration, permet de traiter des zones de messages faiblement

typées comme s'il s'agissait de zones fortement typées. Une zone est faiblement typée si elle peut contenir plusieurs type de données. Une zone est fortement typée si son type et sa structure interne sont connus.

- En phase d'exécution, la primitive de médiation Définition du type de message vous permet de vérifier que le contenu d'un message correspond aux types de données attendus.

Configurateur d'en-tête SOAP
Fournit un mécanisme de gestion des en-têtes dans les messages SOAP.

- Peut créer, définir, copier ou supprimer des en-têtes de messages SOAP.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes SOAP.

Arrêter
Arrête un chemin donné dans le flux sans générer d'exception.

Filtre de type
Permet d'acheminer des messages vers un chemin différent d'un flux, en fonction de leur type.

WebSphere eXtreme Scale Extraire
Vous pouvez extraire les informations depuis un environnement de cache de serveur eXtreme Scale.

- Vous pouvez rechercher des valeurs dans le cache, et les stocker en tant qu'éléments du message à l'aide d'une clé.
- En combinant les primitives de médiation eXtreme Scale Stocker et Extraire, vous pouvez mettre en cache la réponse à partir d'un système dorsal. Les requêtes futures n'auront pas besoin d'accéder à ce système dorsal.
- Vous devez créer des définitions de eXtreme Scale à l'aide de la console d'administration WebSphere ESB, afin de pouvoir indiquer quel serveur eXtreme Scale utiliser.

WebSphere eXtreme Scale Stocker
Vous pouvez stocker les informations dans un environnement de cache de serveur eXtreme Scale.

- Vous pouvez stocker des informations dans un cache eXtreme Scale à l'aide d'une clé et d'un objet.
- En combinant des primitives de médiation deeXtreme Scale Stocker et Extraire, vous pouvez utiliser la primitive de médiation Stocker pour stocker des données dans le cache, et utiliser la primitive de médiation Extraire pour extraire les données précédemment stockées dans le cache.
- Vous devez créer des définitions de eXtreme Scale à l'aide de la console d'administration WebSphere ESB, afin de pouvoir indiquer quel serveur eXtreme Scale utiliser.

Routage dynamique

Vous pouvez acheminer les messages de plusieurs manière à l'aide de points de contact définis en phase d'intégration ou de points de contact déterminés, de manière dynamique, en phase d'exécution.

Le routage dynamique comprend deux cas de routage des messages :

- Le routage dynamique dans lequel le flux est dynamique, mais avec tous les noeuds finaux possibles prédéfinis dans un module SCA (Service Component Architecture).
- Routage de message dans lequel le flux est dynamique, de même que la sélection des points de contact. Les points de contact de service sont sélectionnés à partir d'une source externe, en phase d'exécution.

Sélection de point de contact dynamique

L'environnement d'exécution est doté d'une fonction de routage des messages de demande et de réponse vers une adresse de point de contact identifiée par un élément d'en-tête de message. Cet élément d'en-tête de message peut être mis à jour par des primitives de médiation dans un flux de médiation. L'adresse de point de contact peut être mise à jour avec des informations d'un registre, d'une base de données ou avec

des informations provenant du message même. Le routage des messages de réponse ne s'applique que si la réponse est envoyée par une exportation JAX-WS de service Web.

Pour que l'environnement d'exécution puisse implémenter le routage dynamique sur une demande ou une réponse, la propriété `Utiliser le point de contact dynamique s'il est défini dans l'en-tête de message` doit être définie sur le module SCA. Les développeurs d'intégration peuvent définir la propriété `Utiliser le point de contact dynamique s'il est défini dans l'en-tête de message` ou la promouvoir (la rendre visible en phase d'exécution) de telle sorte que l'administrateur d'exécution puisse la définir. Vous pouvez visualiser les propriétés de module dans la fenêtre Propriétés de module. Pour afficher la fenêtre, cliquez sur **Applications** > **Modules SCA** > **Propriétés de module**. Le développeur d'intégration donne aux propriétés promues des noms d'alias qui sont affichés sur la console d'administration.

Registre

Vous pouvez utiliser IBM WebSphere Service Registry and Repository (WSRR) pour stocker les informations de point de contact de service puis créer les modules SCA pour extraire les points de contact du registre WSRR.

Lorsque vous développez des modules SCA, vous utilisez la primitive de médiation Recherche de noeud final pour permettre au flux de médiation d'effectuer une requête sur un registre WSRR pour un noeud final de service ou un ensemble de noeuds finaux de service. Si un module SCA extrait un ensemble de points de contact, il doit alors utiliser une autre primitive de médiation pour sélectionner le point de contact à privilégier.

Contrôle des règles de médiation des demandes de service

Vous pouvez utiliser des règles de médiation pour contrôler les flux de médiation entre les demandeurs de services et les fournisseurs de services.

Vous pouvez contrôler les flux de médiation en utilisant les règles de médiation stockées dans IBM WebSphere Service Registry and Repository (WSRR). L'implémentation de la gestion des règles de service dans WSRR est basée sur Web Services Policy Framework (WS-Policy).

Pour contrôler les demandes de service en utilisant des règles de médiation, vous devez disposer de modules SCA (Service Component Architecture) et de documents de règles de médiation adaptés dans votre registre WSRR.

Savoir connecter une règle de médiation à une demande de service

Lorsque vous développez un module SCA nécessitant l'utilisation d'une règle de médiation, vous devez inclure une primitive de médiation Résolution de règle dans le flux de médiation. En phase d'exécution, la primitive de médiation Résolution de règle obtient des informations sur la règle de médiation à partir du registre. C'est pourquoi, un module SCA doit contenir un composant de flux de médiation afin de prendre en charge le contrôle des règles de médiation des demandes de service.

Dans le registre, vous pouvez associer une ou plusieurs règles de médiation à un module SCA ou à un service utilisé par le module SCA. Les règles de médiation associées peuvent être utilisées (sont dans la portée) pour tous les messages de service traités par ce module SCA. Les règles de médiation peuvent être associées à des connexions de règles qui définissent des conditions. Les conditions de règle de médiation permettent à différentes règles de médiation de s'appliquer dans différents contextes. En outre, les règles de médiation peuvent comporter des classifications utilisables pour indiquer un état de gouvernance.

WebSphere Service Registry and Repository

WebSphere Service Registry and Repository (WSRR) vous permet de conserver, de consulter et de gérer les informations relatives aux points de contact de service et aux règles de médiation. Vous pouvez utiliser WSRR pour rendre vos applications de service plus dynamiques et plus adaptables à l'évolution des conditions métier.

Introduction

Les flux de médiation peuvent utiliser WSRR comme un mécanisme de recherche dynamique fournissant des informations sur les noeuds de service ou les règles de médiation.

Pour configurer l'accès à WSRR, vous devez créer des documents de définition WSRR via la console d'administration. Vous pouvez aussi utiliser les commandes d'administration WSRR du client de script wsadmin. Les définitions WSRR et leurs propriétés de connexion représentent le mécanisme utilisé pour se connecter à une instance du registre et extraire un point de contact de service ou une règle de médiation.

Noeuds finaux de service

Vous pouvez utiliser WSRR pour conserver des informations sur les services que vous utilisez déjà, que vous prévoyez d'utiliser et dont vous souhaitez avoir connaissance. Ces services peuvent figurer dans vos systèmes ou dans d'autres. Par exemple, une application peut utiliser peut WSRR pour localiser le service le plus à même de répondre à ses besoins fonctionnels et de performances.

Lorsque vous développez un module SCA nécessitant l'accès à des points de contact de service à partir de WSRR, vous devez inclure une primitive de médiation Recherche de point de contact dans le flux de médiation. Pendant la phase d'exécution, celle-ci récupère en effet les points de contact de service du registre.

Règles de médiation

Vous pouvez également utiliser WSRR pour stocker les informations de règle de médiation. Les règles de médiation peuvent vous permettre de contrôler les requêtes de service par substitution dynamique des propriétés du module. Si WSRR contient des règles de médiation associées à un objet représentant votre module SCA ou votre service cible, les règles de médiation peuvent remplacer les propriétés du module. Les conditions de règle de médiation permettent à différentes règles de médiation de s'appliquer dans différents contextes.

Remarque : Les règles de médiation s'occupent du contrôle des flux de médiation mais pas des questions de sécurité.

Lorsque vous développez un module SCA nécessitant l'utilisation d'une règle de médiation, vous devez inclure une primitive de médiation Résolution de règle dans le flux de médiation. En phase d'exécution, la primitive de médiation Résolution de règle obtient des informations sur la règle de médiation à partir du registre.

WebSphere eXtreme Scale

En utilisant le produit WebSphere eXtreme Scale (eXtreme Scale), vous pouvez fournir un système de mise en cache que vous pouvez intégrer à une application IBM Business Process Manager. En utilisant eXtreme Scale avec IBM Business Process Manager, vous pouvez améliorer les temps de réponse du service et la fiabilité, et fournir des fonctionnalités d'intégration supplémentaires.

eXtreme Scale agit comme une grille de données en mémoire élastique et évolutive. La grille de données met en cache, partitionne, réplique et gère les données et la logique métier de façon dynamique sur de nombreux serveurs. Avec eXtreme Scale, vous pouvez également obtenir des qualités de service telles que l'intégrité transactionnelle, la haute disponibilité et des temps de réponse prévisibles.

Vous pouvez utiliser des flux de médiation pour accéder à la fonction de mise en cache de eXtreme Scale, , y compris les primitives de médiation WebSphere eXtreme Scale dans votre flux. Lorsque vous développez un module SCA (Service Component Architecture) nécessitant de stocker des informations dans un cache eXtreme Scale, vous devez inclure une primitive de médiation WebSphere eXtreme Scale Stocker dans le flux de médiation. Si vous souhaitez extraire des informations à partir d'un cache eXtreme Scale, vous devez inclure la primitive de médiation WebSphere eXtreme Scale Extraire. En combinant les deux primitives de médiation dans un flux de médiation, vous pouvez mettre en cache la réponse à partir d'un système dorsal, afin que les demandes futures puissent extraire la réponse du cache.

Pour configurer l'accès à eXtreme Scale , vous devez créer une définition WebSphere eXtreme Scale à l'aide de la console d'administration. Vous pouvez aussi utiliser les commandes d'administration WebSphere eXtreme Scale depuis le client de scriptage wsadmin. Une définition eXtreme Scale est le mécanisme utilisé par les primitives de médiation WebSphere eXtreme Scale Extraire et Stocker pour se connecter à un serveur eXtreme Scale.

Message Service Clients

Message Service Clients est disponible pour C/C++ et .NET pour permettre aux applications non-Java de se connecter au bus de service d'entreprise.

Message Service Clients for C/C++ and .NET fournit l'API XMS qui dispose du même groupe d'interfaces que l'API Java Message Service (JMS). Message Service Client for C/C++ contient deux implémentations de XMS, une utilisée par les applications C et une autre utilisée par les applications C++s. Message Service Client for .NET contient une implémentation totalement gérée de XMS, qui peut être utilisée par n'importe quel langage compatible .NET.

Vous pouvez obtenir Message Service Clients for .NET à l'adresse suivante : http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en

Vous pouvez obtenir Message Service Clients for C/C++ à l'adresse suivante : http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Vous pouvez également installer et utiliser le support client Java EE de WebSphere Application Server Network Deployment, y compris les client de services Web, EJB et JMS.