IBM WebSphere Process Server for Multiplatforms

Fiches techniques

Version 7.0.0

juin 2012

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. Les informations qui y sont fournies sont susceptibles d'être modifiées avant que les produits décrits ne deviennent eux-mêmes disponibles. En outre, il peut contenir des informations ou des références concernant certains produits, logiciels ou services non annoncés dans ce pays. Cela ne signifie cependant pas qu'ils y seront annoncés.

Pour plus de détails, pour toute demande d'ordre technique, ou pour obtenir des exemplaires de documents IBM, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial.

Vous pouvez également consulter les serveurs Internet suivants :

- http://www.fr.ibm.com (serveur IBM en France)
- http://www.can.ibm.com (serveur IBM au Canada)
- http://www.ibm.com (serveur IBM aux Etats-Unis)

Compagnie IBM France
Direction Qualité
17, avenue de l'Europe
92275 Bois-Colombes Cedex

Table des matières

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.

OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
(Pos1)		Home
Fin	Fin	End
(PgAr)		PgUp
(PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
(Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

Fiches techniques

Les informations contenues dans les fiches techniques présentent les normes et les aspects techniques liés à l'architecture produit.

Service Component Architecture

L'architecture SCA (Service Component Architecture) permet une architecture orientée services, proposée par de nombreuses sociétés, dont IBM. SCA est une plateforme et un modèle de programmation indépendant du fournisseur qui offre des moyens simples et cohérents d'exprimer la logique métier et les données métier sous forme de services SOA, quels que soient les détails d'implémentation techniques. Dans cette section, nous examinons les services SCA et les objets de données.

SCA et appel de service

Si vous considérez les trois aspects composant un modèle de programmation (données, appel et composition) et que vous appliquez les nouveaux paradigmes d'une approche orientée services, le nouveau modèle de programmation d'une architecture SOA commence à prendre forme. SCA (Service Component Architecture) permet d'appeler des services métier dans des solutions SOA.

Les constructions architecturales qui constituent une architecture orientée services incluent une méthode de représentation des données qui est échangée entre les services, un mécanisme d'appel des services et une méthode de composition de services dans des applications métier intégrées de plus grande taille. Actuellement, il existe de nombreux modèles de programmation permettant de prendre en charge chacune d'elles. Cette situation représente un double défi pour les développeurs : ils doivent non seulement résoudre un incident métier particulier, mais également choisir et comprendre la technologie d'implémentation appropriée. L'un des objectifs importants de la solution SOA de WebSphere Process Server est de mitiger ces complexités. Pour cela, les divers modèles de programmation utilisés pour l'implémentation d'applications métier orientées services sont regroupés dans un modèle de programmation simplifié.

Cette section s'intéresse plus spécifiquement à SCA (Service Component Architecture) dans WebSphere Process Server comme modèle de composant orienté service permettant de définir et d'appeler des services métier. SCA joue un rôle important en offrant un modèle d'appel à la solution SOA de WebSphere Process Server. SCA joue également un rôle dans la composition de services métier en applications métier composites.

Tout d'abord, nous constatons que le langage XML (Extensible Markup Language) est principalement utilisé pour représenter des données et que leur programmation est effectuée à l'aide d'objets métier basés sur la spécification SDO (Service Data Object) ou de fonctions XML natives, telle que XPath ou XSLT (Extensible Stylesheet Language Transformation). Ensuite, un appel de service effectue un mappage vers l'architecture SCA (Service Component Architecture). Pour finir, la composition est intégrée à l'orchestration des processus à l'aide du langage BPEL (Business Process Execution Language). Le schéma suivant illustre les trois aspects de ce nouveau modèle de programmation.

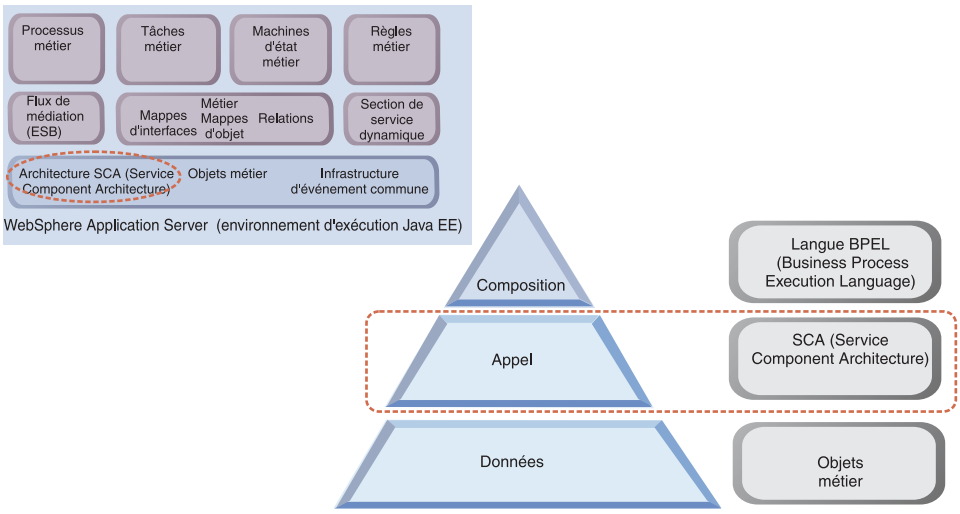


Figure 1. Représentation des données, des appels et des compositions dans un modèle de programmation pour SOA

SCA est destiné à offrir un modèle de programmation simplifié pour créer des applications qui fonctionnent dans un environnement d'exécution Java EE. Il repose sur des concepts et des techniques qui sont des améliorations de la technologie Java existante. L'un des aspects importants de SCA est d'activer la séparation entre la logique métier de l'implémentation et les détails de l'implémentation. Pour accomplir cela, SCA offre une abstraction unique pour les types de service qui peuvent être déjà exprimés sous forme de beans session, services Web, classes Java ou BPEL. La possibilité de séparer la logique métier de la logique d'infrastructure est importante pour réduire le nombre de ressources informatiques requises pour générer une application d'entreprise et donne davantage de temps au développeurs pour résoudre un incident métier particulier plutôt que de se concentrer sur la détermination de la technologie d'implémentation à utiliser.

Modules

Un *module* est une unité de déploiement qui détermine quels artefacts sont intégrés ensemble dans un fichier EAR (Enterprise Archive). Les composants d'un module sont colocalisés pour des raisons de performances et peuvent transmettre leurs données par référence. Un module peut être considéré comme un mécanisme de configuration ; il définit une limite organisationnelle pour les artefacts.

Un module est composé de composants de service, d'importations et d'exportations. Les composants de service, les importations et les exportations résident dans le même projet et de même dossier racine, qui contient également la connexion qui associe les composants et les liaisons nécessaires pour les importations et les exportations. Un module peut également contenir les implémentations et les interfaces référencées par ses composants, importations et exportations ou ces derniers peuvent être placés dans d'autres projets, tels qu'un projet de bibliothèque.

Il existe deux types de module. Le premier est un module appelé *module* (ou parfois module d'intégration métier) qui contient un choix de nombreux types de composant, souvent utilisés pour prendre en charge un processus métier. Le second est un module appelé *module de médiation*, qui contient jusqu'à un composant, un ou plusieurs composants de flux de médiation et aucun, un ou plusieurs composants Java qui étendent le composant de flux de médiation.

Un module peut contenir un ou plusieurs composants de flux de médiation.

Pourquoi existe-t-il deux types de module ? Le premier type de module est principalement conçu pour les processus métier. Un module de médiation est similaire à une passerelle vers des services externes existants, qui est courante dans les architectures de bus de service d'entreprise. Ces services externes ou exportations sont accessibles dans un module de médiation par des importations ou des fournisseurs de services. En dissociant les demandeurs de service client des fournisseurs de services à l'aide d'un flux de médiation, vos applications gagnent en flexibilité et résilience, un objectif de l'architecture orientée services. Par exemple, votre flux de médiation peut consigner les messages entrants, acheminer des messages à un service spécifique déterminé lors de la phase d'exécution ou transformer des données pour pouvoir les transmettre à un autre service. Ces fonctions peuvent être ajoutées et modifiées dans le temps sans modifier les services du demandeur ou du fournisseur.

Un module génère une application de service testée et déployée sur WebSphere Process Server. Un module de médiation génère une application de service testée et déployée sur le serveur WebSphere Process Server ou WebSphere Enterprise Service Bus. Ces deux types de module prennent en charge les importations et les exportations.

Les implémentations, les interfaces, les objets métier, les mappes d'objet métier, les rôles, les relations et autres artefacts doivent souvent être partagés entre les modules. Une *bibliothèque* est un projet utilisé pour stocker ces ressources partagées.

Dans la figure 2, à la page 4, le module contient deux composants de service, chacun contenant une implémentation. Le module contient également les interfaces appropriées et les références requises par les composants de service. Le deuxième composant de service ne contient pas de référence car il n'appelle pas de service externe.

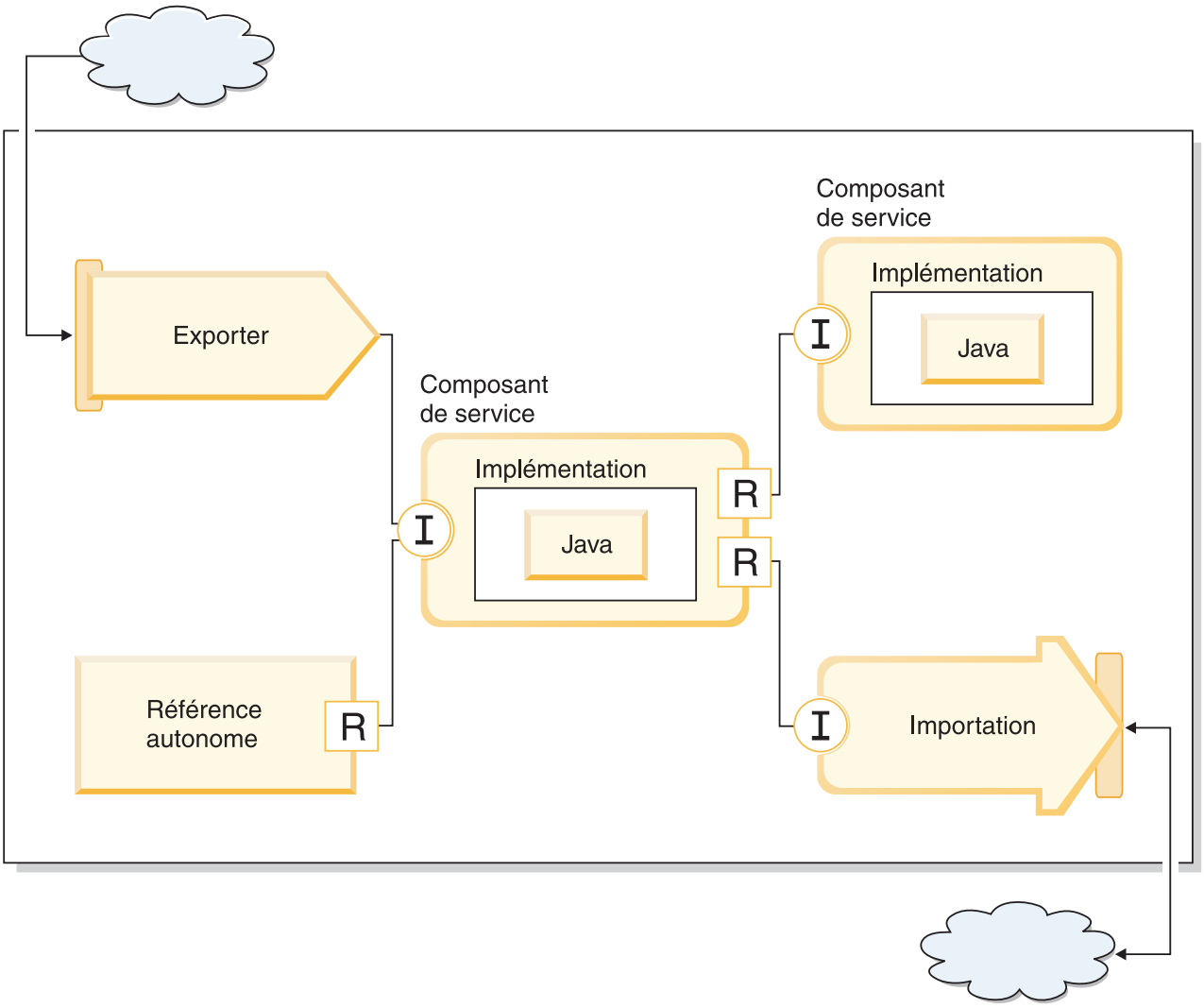
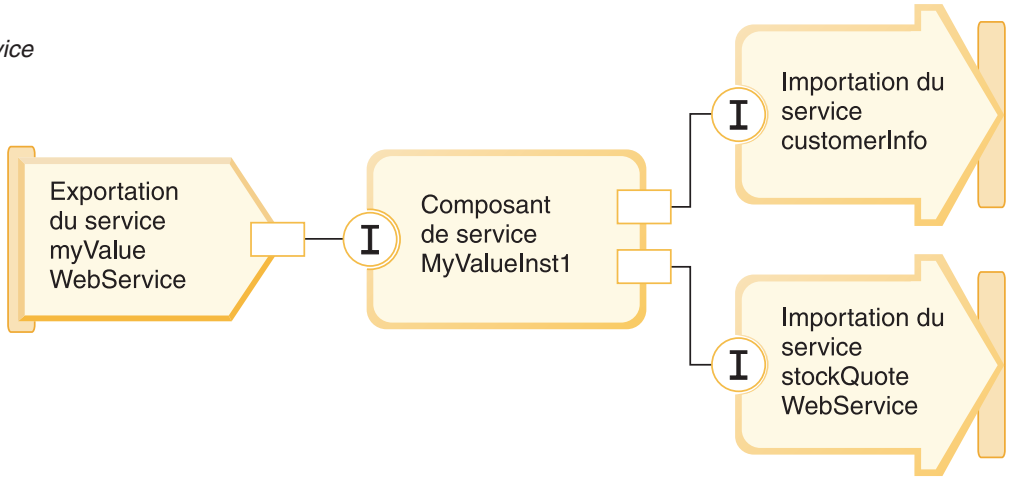


Figure 2. Structure d'un module

Dans figure 3, le module contient une exportation, deux importations et un composant de service qui les utilise. Une connexion relie les interfaces et les références.

Figure 3. Module de service



Les artefacts de module et de module de médiation incluent les éléments suivants :

- Définition de module - Définit le module.
- Composants de service - Définitions des services dans le module. Un nom de composant de service dans un module est unique. Toutefois, un composant de service peut posséder un nom d'affichage arbitraire, qui correspond généralement à un nom plus utile pour un utilisateur.
- Importations - Définitions des importations, qui sont des appels à des services externes à ce module. Les importations possèdent des liaisons, qui sont abordées dans la section Liaisons.
- Exportations - Définitions des exportations, qui permettent d'exposer les composants à des demandeurs externes à ce module. Les exportations possèdent des liaisons, qui sont abordées dans la section Liaisons.
- Références - Références d'un composant à un autre dans le module.
- Références autonomes - Références à des applications non définies comme composants SCA (Service Component Architecture), telles que des pages JavaServer, qui permettent à ces applications d'interagir avec des composants SCA (Service Component Architecture). Il ne peut exister qu'un seul artefact de références autonomes par module.
- Autres artefacts : ces artefacts incluent les fichiers WSDL, les classes Java, les fichiers XSD, les processus BPEL, etc.

Composants de service

Un composant de service configure une implémentation de service. Un composant de service est présenté dans un diagramme de bloc standard.

En plus de fournir une syntaxe cohérente et un mécanisme d'appel des services, SCA (Service Component Architecture) sert de cadre d'appel et permet aux développeurs d'encapsuler les implémentations de services dans des composants réutilisables. SCA permet aux développeurs de définir des interfaces, des implémentations et des références indépendamment de la technologie utilisée. Cette approche vous donne la possibilité d'associer des éléments à la technologie de votre choix. L'architecture SCA distingue la logique métier de l'infrastructure afin que les programmeurs d'application puissent se consacrer à la résolution de problèmes métier.

Un composant se compose d'une implémentation, qui est masquée lors de l'utilisation des outils de WebSphere Integration Developer, d'une ou plusieurs interfaces, qui définissent ses entrées, ses sorties et incidents, ainsi que d'une ou plusieurs références. Une référence identifie l'interface d'un autre service ou composant requis ou consommé par ce composant. Une interface peut être définie dans l'une des deux langues suivantes : un type de port WSDL ou Java. Une interface prend en charge les interactions de style synchrone et asynchrone. L'implémentation d'un composant peut être effectuée dans diverses langues.

Le type d'interface recommandée est WSDL et nos tutoriels et exemples utilisent toujours le type d'interface WSDL. Toutefois, une interface Java est prise en charge et utilisée principalement lors de l'importation d'un EJB de session sans état. Si vous développez un composant Java descendant (vous définissez un composant et ajoutez l'implémentation Java par la suite), vous devez quand même utiliser une interface WSDL. Vous ne pouvez pas mélanger les composants basés sur une interface WSDL avec ceux basés sur une interface Java.

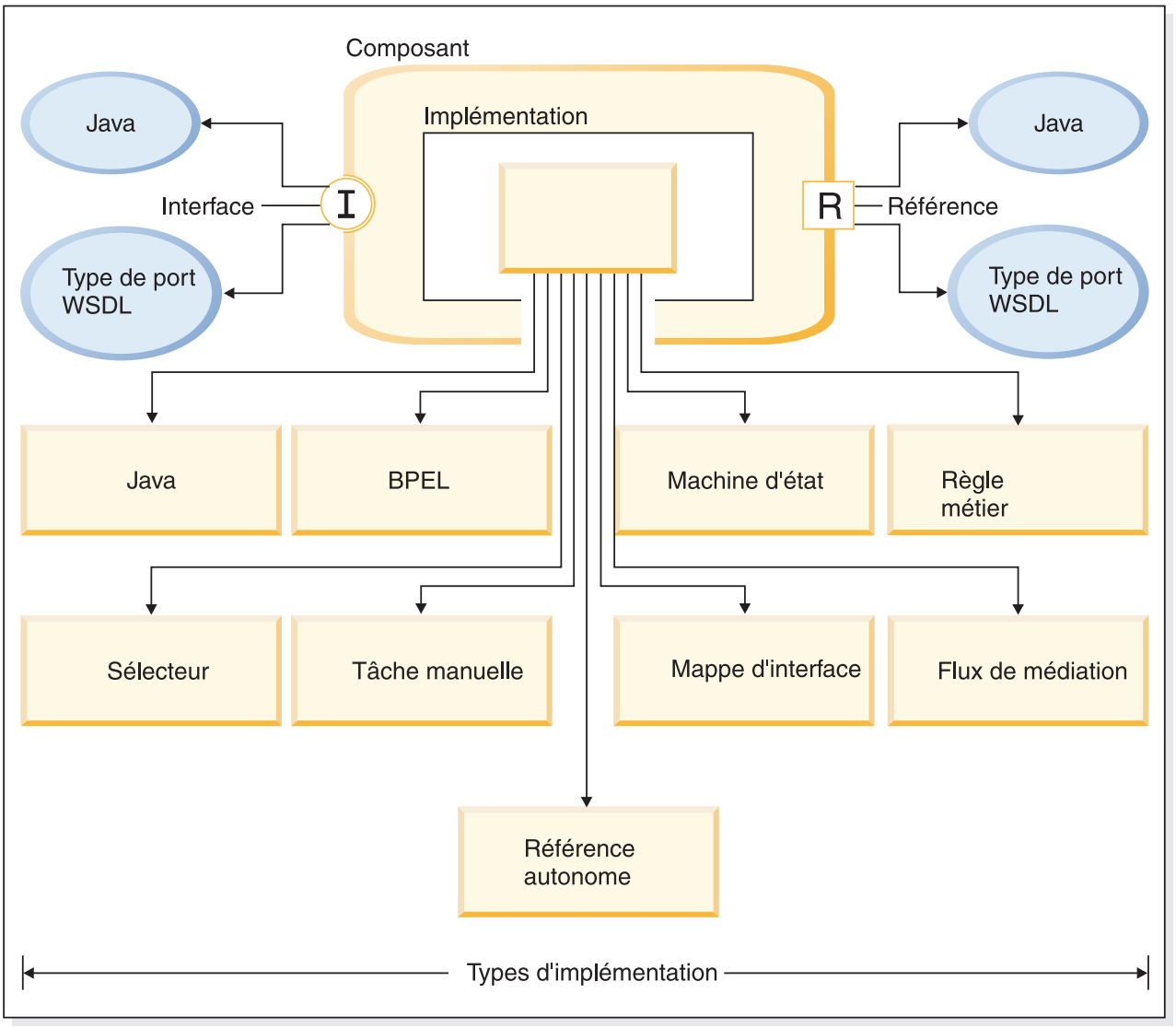


Figure 4. Structure d'un composant

Dans figure 5, à la page 7, un composant se trouve au centre. Son implémentation, MyValueImpl, est en Java, comme son interface. Il possède deux références : une autre interface Java et une interface WSDL.

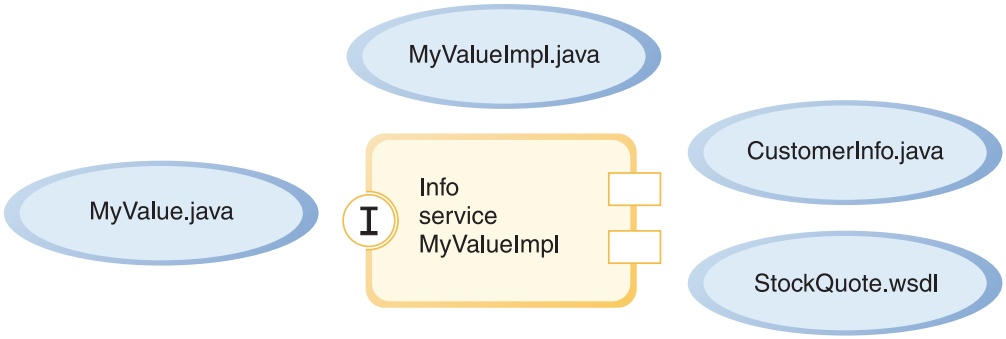


Figure 5. Structure d'un composant de service

Lorsque vous utilisez ce composant, comme illustré ci-après, vous ne voyez en réalité que le composant lui-même. Une référence à ce composant à partir d'un autre composant est représentée graphiquement par une ligne le reliant à son interface. Une référence à ce composant est représentée par une ligne entre son point de référence et l'interface de l'autre composant. Une référence représente un service consommé par ce composant. En désignant une référence et en ne spécifiant que son interface, l'auteur de l'implémentation de composant peut reporter la liaison de cette référence à un service réel. Le spécialiste d'intégration effectuera cette liaison par la suite en connectant la référence à l'interface d'un autre composant ou à une importation. Cette faible association, qui permet le report des liaisons et la réutilisation des implémentations, est l'une des principales raisons pour laquelle l'architecture SCA de WebSphere Integration Developer est utilisée.

Un composant peut également posséder des propriétés et des qualificateurs. Un qualificateur est une instruction QoS (qualité de service) sur les interfaces et les références pour la phase d'exécution.

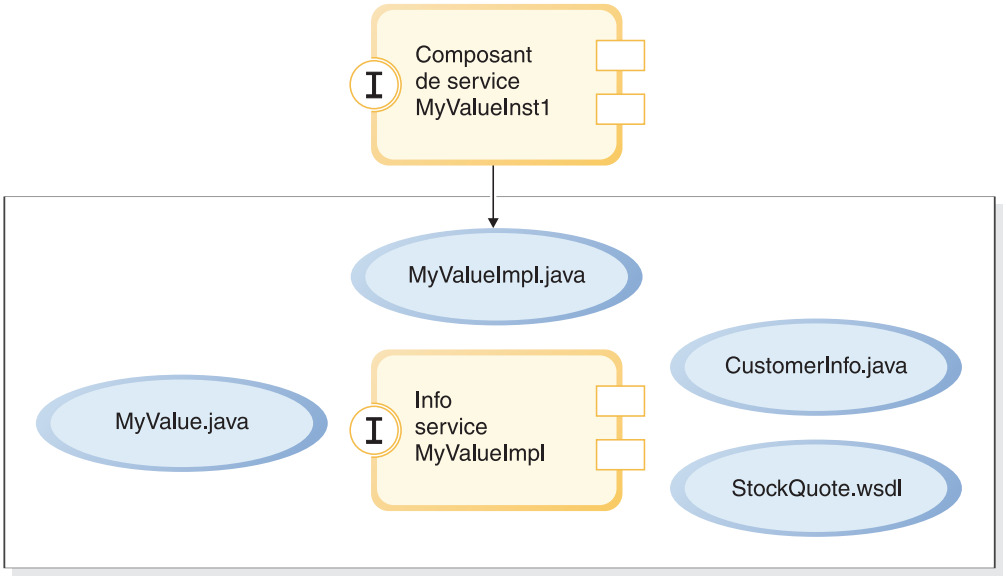


Figure 6. Instance d'un composant de service

Les types d'implémentation de composant de service correspondent aux implémentations des composants de service.

WebSphere Integration Developer prend en charge les artefacts d'implémentation de WebSphere Process Server et WebSphere Enterprise Service Bus :

Tableau 1. Artefacts d'implémentation

WebSphere Process Server	**WebSphere Enterprise Service Bus**
Objets Java	Objets Java
Processus métier	Flux de médiation
Machines d'état métier	
Règles métier	
Sélecteurs	
Tâches manuelles	
Mappes d'interfaces	
Flux de médiation	

Remarque : Remarque : Les mappes d'interface sont dépréciées à partir de WebSphere Process Server, version 7.0. Vous pouvez migrer vos composants de mappe d'interface existants dans WebSphere Integration Developer pour utiliser les fonctions du composant de flux de médiation.

Les implémentations de composant standard des services sont décrites dans les rubriques de cette section. Ces implémentations apparaissent dans des services, dans l'éditeur d'assemblage et/ou les processus BPEL.

Objets Java

Une implémentation d'un composant dans Java est appelée objet Java.

Une implémentation courante est un composant écrit en Java. Cette implémentation est parfois surnommée "objet Java simple". Généralement, cette implémentation possède un type d'interface WSDL, mais elle peut également posséder une interface Java. Si plusieurs interfaces sont spécifiées, vous ne pouvez pas mélanger les interfaces WSDL et les interfaces Java. Vous pouvez toutefois "joindre" une application créée avec un ensemble d'interfaces WSDL à une application contenant un ensemble d'interfaces Java. Pour cela, vous pouvez consulter l'un des exemples de la galerie d'exemples dans la vue Bienvenue.

Lorsque vous utilisez un objet Java, le code reste masqué dans le contexte des éditeurs.

Un objet Java peut être utilisé dans un module de médiation. Il peut être déployé sur un serveur WebSphere Process Server ou WebSphere Enterprise Service Bus.

Processus BPEL

Un composant *Processus BPEL* implémente un processus métier.

Son langage d'implémentation est la norme BPEL4WS (Business Process Execution Language for Web Services) et ses extensions IBM. Un processus BPEL implémente un service à exécution potentiellement longue via l'utilisation de services plus élémentaires. Un processus BPEL créé dans l'éditeur de processus peut effectuer les opérations suivantes :

- Décrire l'orchestration des autres services à l'aide de graphiques de flux de contrôle
- Utiliser des variables pour conserver l'état des processus

- Utiliser un traitement d'erreurs sophistiqué via la gestion des incidents
- Prendre en charge les événements asynchrones
- Effectuer une corrélation des demandes entrantes avec l'instance appropriée d'un processus particulier en utilisant des jeux de corrélations pour marquer ces données métier dans la demande qui identifie l'instance (par exemple, un ID client)
- Fournir des transactions étendues via un support de compensation sophistiqué

En plus de ces éléments BPEL standard, WebSphere Integration Developer étend également BPEL pour inclure des personnes dans un processus en prenant en charge les *tâches manuelles*. Par exemple, cette extension peut ajouter à un processus la condition qu'une personne approuve un prêt.

L'éditeur de processus utilise des représentations visuelles de constructions BPEL pour générer vos processus métier rapidement et simplement.

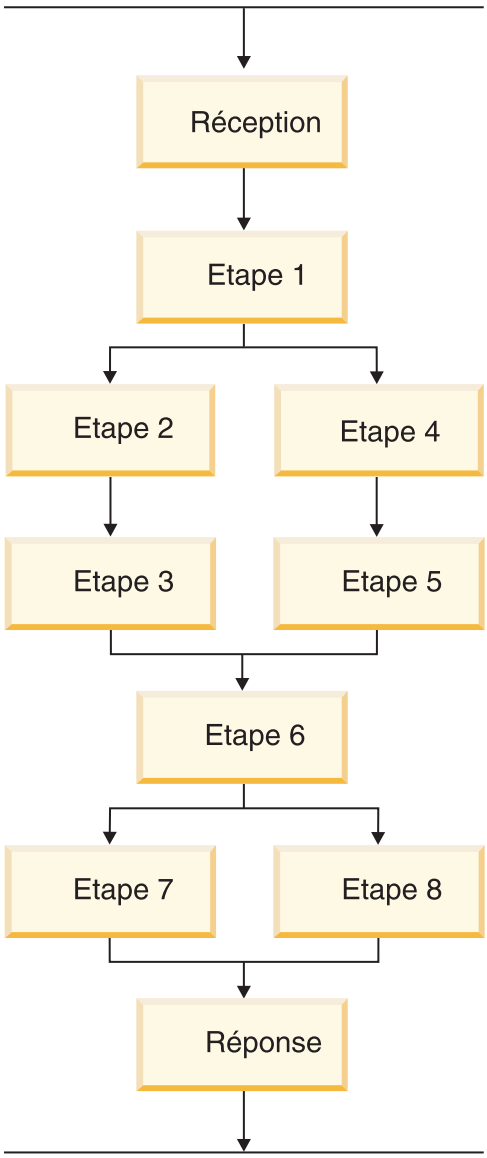


Figure 7. Processus métier simple

Un processus métier ne peut pas être utilisé dans un module de médiation. Elle ne peut être déployée que sur un serveur WebSphere Process Server.

Machines d'état

Une machine d'état représente une autre manière de créer un processus métier. Une machine d'état convient aux processus relatifs aux états changeants et non à un flux de contrôle. Un état définit ce que peut faire un artefact à un moment précis. Une *machine d'état* est une implémentation de cet ensemble d'états.

Les machines d'état représentent un moyen courant d'afficher un ensemble d'états interdépendants dans un processus. Un distributeur de boissons constitue une machine d'état répandue. Vous insérez des pièces dans la machine et récupérez la monnaie exacte, avec si possible votre boisson, car la machine d'état répartit mécaniquement les pièces à vous rendre en fonction de celles que vous avez insérées. Dans le diagramme ci-après, une machine d'état type est illustrée comme créée par l'éditeur de machine d'état. Dans la machine d'état, un article est acheté et expédié à un client.

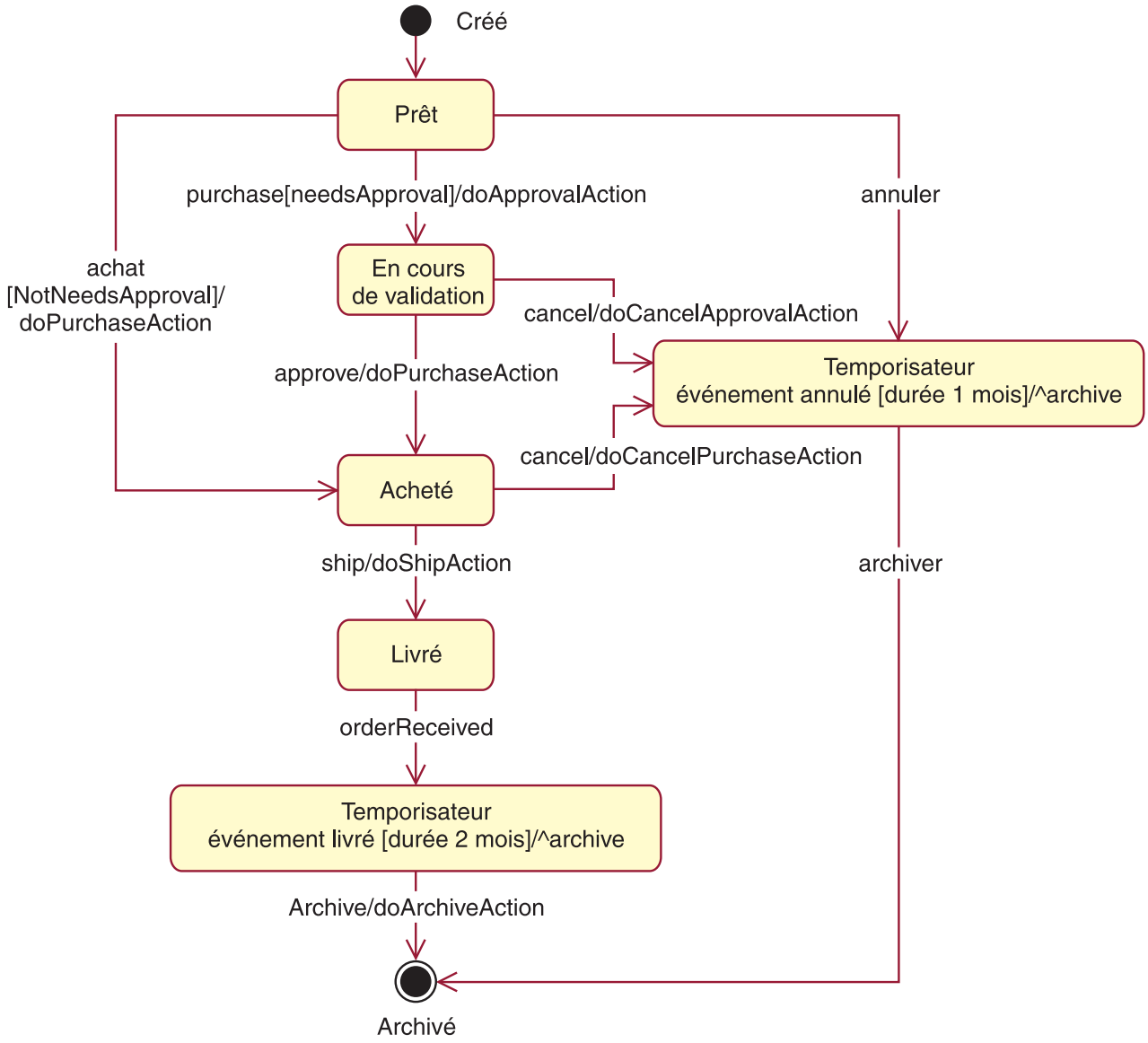


Figure 8. Machine d'état

Une machine d'état ne peut pas être utilisée dans un module de médiation. Elle ne peut être déployée que sur un serveur WebSphere Process Server.

Règles métier

Les règles métier viennent en complément des processus métier et des machines d'état. S'il existe une condition avec une variable, par exemple, une *règle métier* peut modifier la valeur de cette variable lors de la phase d'exécution. Créée par un langage de programmation visuelle, une règle métier prend une décision en fonction du contexte. Cette décision peut-être simple ou complexe. Les règles métier ne sont pas procédurales et peuvent être modifiées indépendamment d'une application.

Les règles métier déterminent le résultat d'un processus en fonction d'un contexte. Les règles métier sont utilisées dans les situations professionnelles quotidiennes pour prendre une décision en fonction d'un ensemble de circonstances donné. Cette décision peut nécessiter de nombreuses règles pour couvrir toutes les circonstances. Les règles métier dans un processus métier permettent aux

applications de répondre rapidement aux variations auxquelles est sujette votre entreprise. Dans une compagnie d'assurance, la règle suivante peut être appliquée pour déterminer si la voiture d'un client potentiel peut être assurée : *Si le souscripteur est un homme de plus de 25 ans, qu'il possède une voiture de sport et qu'il a été assuré par notre compagnie depuis 5 ans, il peut être assuré moyennant le versement d'une cotisation mensuelle de 100 euros.*

WebSphere Integration Developer offre un certain nombre d'approches pour créer des règles métier. Vous pouvez créer des règles if-then ou des tables de décision, qui forment toutes le résultat de votre processus. Ces règles sont indépendantes du processus, ce qui signifie que vous pouvez les modifier à tout moment sans avoir à réeffectuer votre processus. Par exemple, en fonction de l'emplacement de votre société, une règle peut spécifier : *Si la date est comprise entre le 26 décembre et le 1er janvier, offrez une réduction de 20 %*. Toutefois, si les ventes restent faibles, vous pouvez à tout moment augmenter cette remise à 40 %.

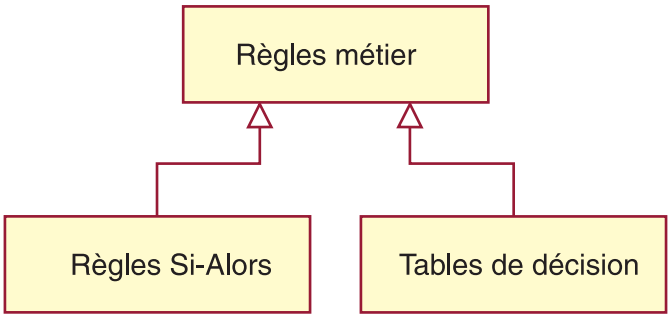


Figure 9. Types de règle métier

Les règles métier ne peuvent pas être utilisées dans un module de médiation. Elles ne peuvent être déployées que sur un serveur WebSphere Process Server.

Sélecteurs

Les applications intégrées peuvent interagir de plusieurs manières. Un *sélecteur* est utilisé pour acheminer une opération d'une application client vers un ou plusieurs composants possibles en vue de son implémentation.

L'acheminement vers un composant est basé sur des dates. Voici, par exemple, une route basée sur une date : *Deux semaines avant la rentrée scolaire, proposer un prix spécial sur les produits scolaires.* Les entreprises possèdent de nombreuses routes basées sur des dates. Un sélecteur décide de choisir une route plutôt qu'une autre lors de la phase d'exécution, en fonction d'une date. Par exemple, dans le cadre d'une période précédant la rentrée scolaire, l'offre indiquée plus haut sera choisie. Toutefois, s'il s'agit de la fin de l'année scolaire, une offre peut être proposée pour aider les enfants à la saison estivale.

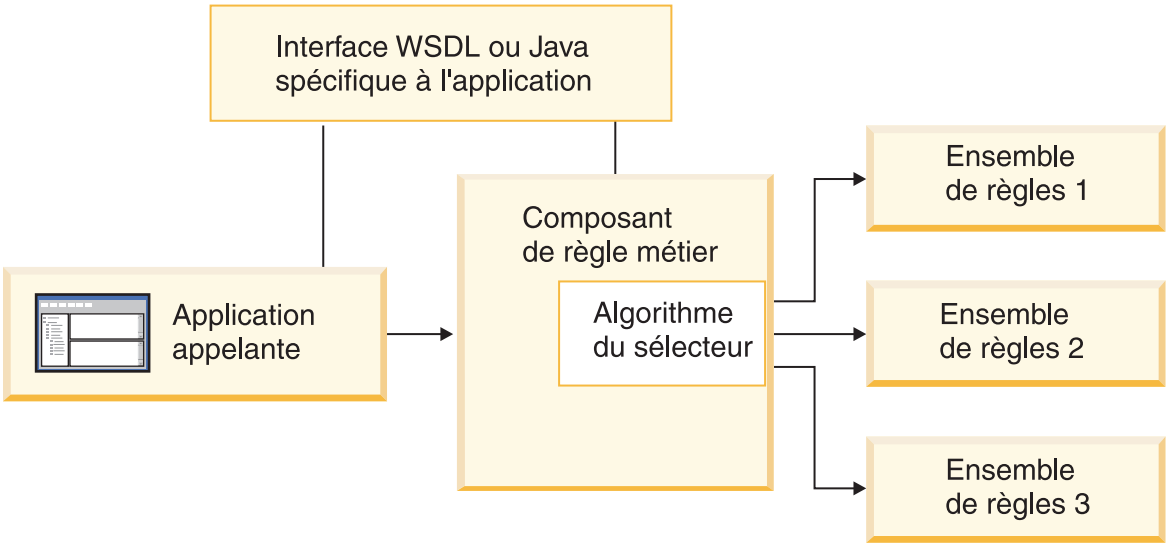


Figure 10. Sélection dans un ensemble de règles métier

Un sélecteur ne peut pas être utilisé dans un module de médiation. Il ne peut être déployé que sur un serveur WebSphere Process Server.

Tâche manuelle

Un composant de *tâche manuelle* implémente une tâche effectuée par une personne. Il représente l'implication d'une personne dans un processus métier.

Des personnes ont parfois besoin d'intervenir dans un processus métier. Par exemple, un client souhaite acheter un article dont le prix est supérieur à sa limite de crédit. Une tâche manuelle vous permet d'intervenir et de substituer une règle métier qui empêche le client d'effectuer l'achat. Une tâche manuelle peut avoir des attributs, tels que la définition du propriétaire de la tâche et l'offre d'un processus d'escalade au cas où la personne spécifiée n'est pas disponible. Le composant de tâche manuelle reconnaît le fait que de nombreux processus requièrent une humaine pour des tâches telles que les vérifications, les recherches et les approbations.

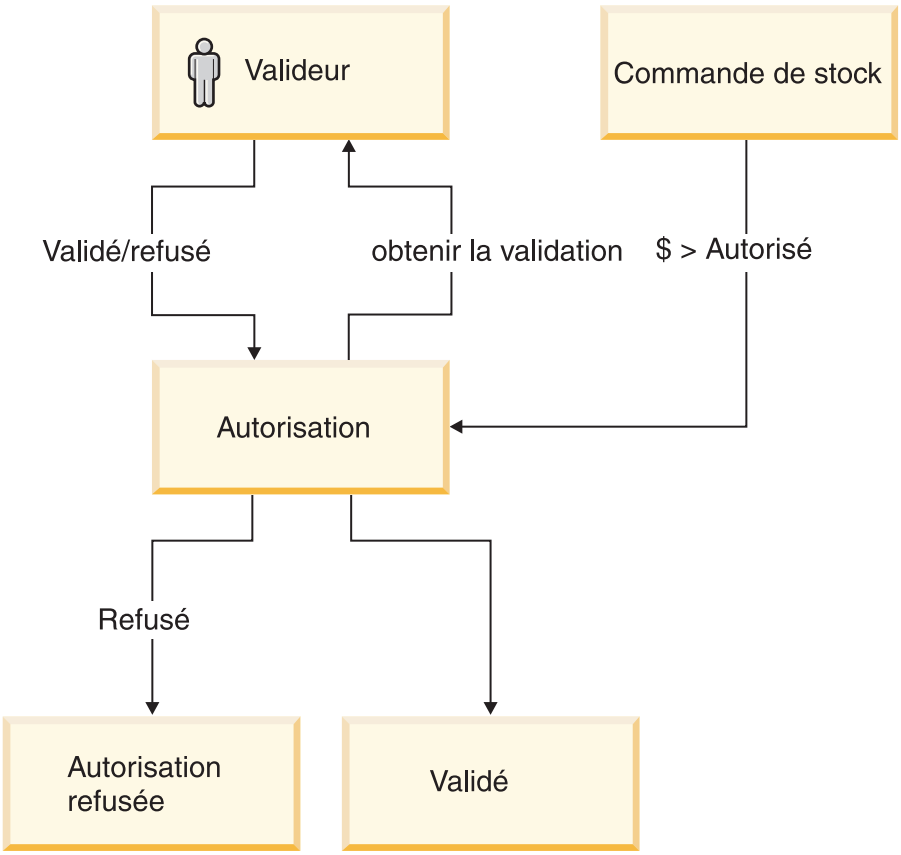


Figure 11. Composant de tâche manuelle

Une tâche manuelle ne peut pas être utilisée dans un module de médiation. Elle ne peut être déployée que sur un serveur WebSphere Process Server.

Mappe d'interface

Une *mappe d'interface* résout les différences entre les interfaces de composants en interaction.

Remarque : Les mappes d'interface sont dépréciées à partir de WebSphere Process Server, version 7.0. Vous pouvez migrer vos composants de mappe d'interface existants dans WebSphere Integration Developer pour utiliser les fonctions du composant de flux de médiation.

Les différences entre les interfaces des composants qui doivent interagir entre eux sont courantes. En effet, dans WebSphere Integration Developer, vous assemblez souvent des composants créés pour des applications différentes. Le fait de pouvoir les réutiliser pour créer une application représente l'une des forces de WebSphere Integration Developer, car cela vous évite de recoder des composants similaires. Cependant, vous devez généralement y apporter de légères modifications.

Par exemple, deux composants peuvent contenir des méthodes qui effectue à peu près la même action, mais qui possèdent des noms différents, tels que getCredit et getCreditRating. Ils peuvent également posséder des noms d'opération différents et les opérations peuvent contenir des types de paramètre différents. Une mappe d'interface mappe les opérations et les paramètres de ces méthodes de sorte que les différences soient résolues et que les deux composants puissent interagir. Une

mappe d'interface se présente comme une passerelle entre les interfaces de deux composants et permet ainsi de connecter ces derniers en dépit de leurs différences.

Une mappe d'interface existe indépendamment des composants qui l'utilisent, ce qui signifie qu'il n'est pas nécessaire de modifier les composants eux-mêmes.

Une mappe d'interface ne peut pas être utilisée dans un module de médiation. Il ne peut être déployé que sur un serveur WebSphere Process Server.

Flux de médiation

La *médiation* est un moyen de servir d'intermédiaire ou d'intervenir de manière dynamique entre les services. Un *flux de médiation* implémente une médiation.

La médiation possède plusieurs fonctions utiles. Par exemple, vous pouvez utiliser la médiation lorsque vous devez transformer des données d'un service dans un format acceptable pour un service ultérieur. La consignation permet de consigner les messages d'un service avant de les envoyer au service suivant. Le routage permet d'acheminer les données d'un service dans un service approprié déterminé par le flux de médiation. Une médiation fonctionne indépendamment des services auxquels elle se connecte. Une médiation dans l'éditeur d'assemblage apparaît comme composant de flux de médiation entre les exportations et les importations.

Dans le diagramme qui suit, trois demandeurs de service ou exportations envoient leurs données en sortie à l'interface du composant de flux de médiation. Le composant de flux de médiation achemine ensuite les données appropriées à deux fournisseurs de services ou importations.

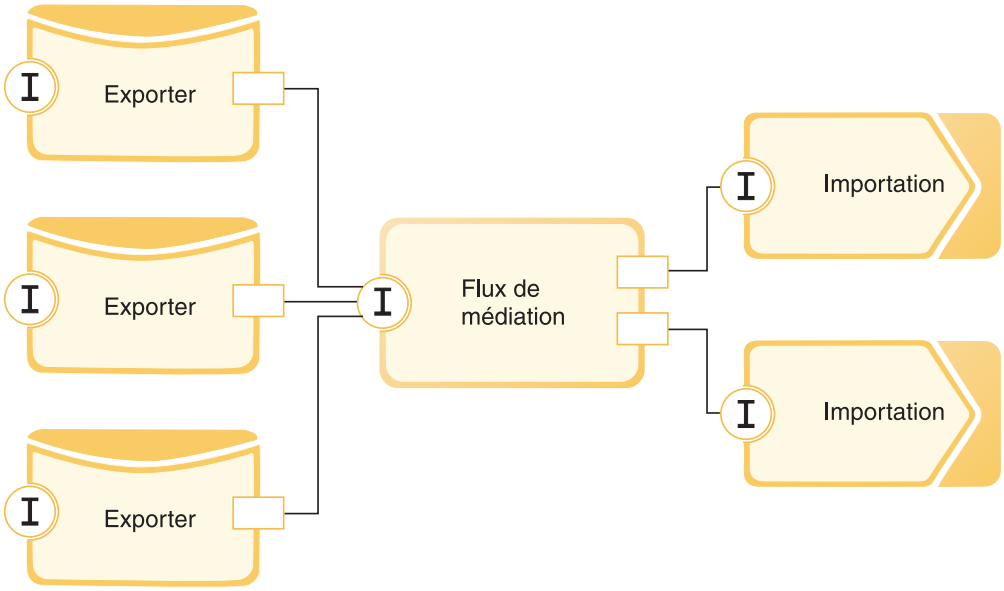


Figure 12. Composant de flux de médiation entre trois demandeurs de service ou exportations et deux fournisseurs de services ou importations

Un flux de médiation est une construction similaire à un flux créée à l'aide de l'éditeur de flux de médiation. Lorsque vous sélectionnez un composant de flux de médiation dans l'éditeur d'assemblage, l'éditeur de flux de médiation est lancé. Dans l'éditeur de flux de médiation, une opération d'un service, le demandeur de service ou l'exportation, est mappée à l'opération d'un autre service, le fournisseur de services ou l'importation, avec les fonctions fournies par l'éditeur de flux de

médiation. Ces fonctions sont appelées *primitives de médiation* et sont connectées dans un flux de médiation comme illustré dans le diagramme ci-après. Les primitives de médiation sont fournies par IBM, mais vous pouvez également créer des primitives personnalisées. Les primitives de médiation peuvent agir sur le contenu du message et son contexte, le contexte correspondant aux informations spécifiques à la liaison, telles que les en-têtes SOAP ou JMS ou les propriétés définies par l'utilisateur.

Dans le diagramme qui suit, une opération appelée applyforLoan envoie d'abord un message à une primitive de consignation, Journal, qui enregistre le message. La primitive Journal envoie le message à la primitive Filtre, qui, en fonction du message, achemine ce dernier à une opération processBusinessLoan ou processPersonalLoan.

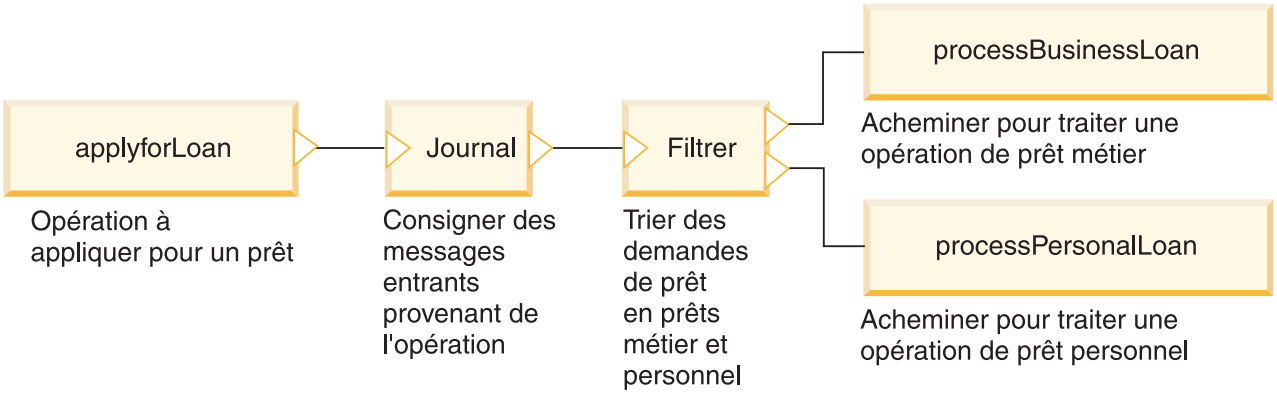


Figure 13. Flux de médiation entre des opérations

Comme expliqué dans la section Modules, les flux de médiation peuvent se trouver dans un module ou un module de médiation. Ces deux types de module peuvent contenir un ou plusieurs composants de flux de médiation, ainsi que zéro, un ou plusieurs composants Java qui étendent le composant de flux de médiation. Un module peut être déployé sur un serveur WebSphere Process Server. Un module de médiation peut être déployé sur un serveur WebSphere Process Server ou WebSphere Enterprise Service Bus.

Qualifiants de service

Une application communique ses besoins en matière de qualité de service à l'environnement d'exécution en spécifiant des *qualifiants de service*. Ces derniers régissent l'interaction entre un client de service et un service cible.

Des qualifiants peuvent être spécifiés dans les références de composant de service, les interfaces et les implémentations. La déclaration des valeurs de qualité de service (QoS) étant externe à une implémentation, vous pouvez modifier ces valeurs sans modifier l'implémentation ou les définir différemment lorsque plusieurs instances d'une même implémentation sont utilisées dans des contextes différents.

Les différentes catégories de qualifiants sont les suivantes :

- Transaction - Règles du type de transaction
- Session d'activité - Règles de connexion à la session active
- Sécurité - Règles des droits d'accès
- Fiabilité asynchrone - Règles de distribution de messages asynchrones

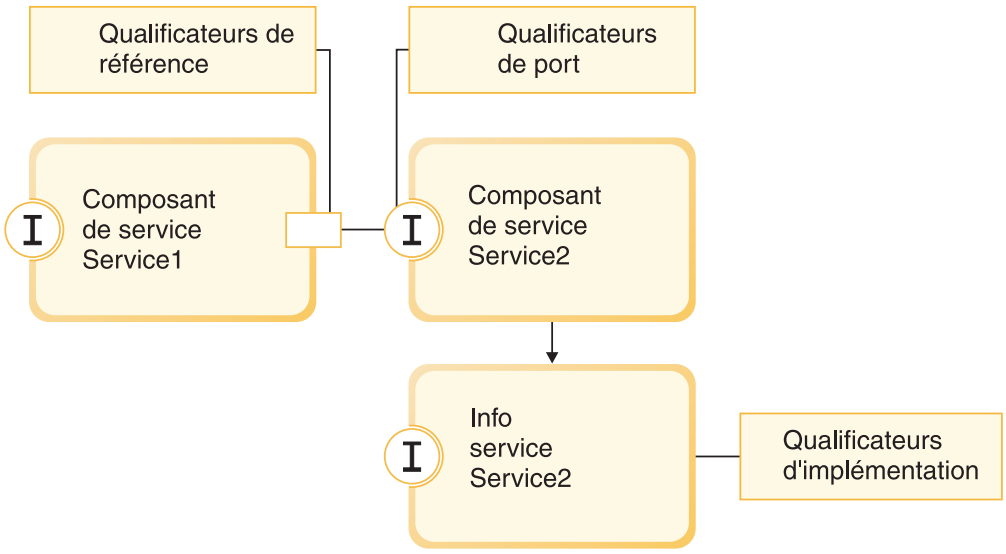


Figure 14. Qualificatifs

Références autonomes

Les *références autonomes* sont des références à des applications non définies comme composants SCA (Service Component Architecture). Par exemple, des pages JavaServer ou des servlets. Les références autonomes permettent à ces applications d'interagir avec des composants SCA (Service Component Architecture).

Les références autonomes ne possèdent ni interface, ni implémentation (car l'implémentation se trouve en dehors de la portée du module). Un module peut ne comporter aucune référence autonome ou contenir un artefact de références autonomes. Les références autonome ont cet aspect pratique en cela qu'elles vous permettent d'utiliser vos applications existantes avec des composants SCA (Service Component Architecture) créés dans WebSphere Integration Developer.

Les références autonomes peuvent être utilisées dans un module de médiation. Elles peuvent être déployées sur un serveur WebSphere Process Server ou WebSphere Enterprise Service Bus.

Objets métier

Les objets métier complètent SCA (Service Component Architecture). Service Component Architecture définit les services comme des composants et la connectivité entre ces services. Les *objets métier* définissent les données circulant entre les composants.

Chaque composant transmet des informations comme entrées et sorties. Lorsqu'un service est appelé, les objets de données sont transmis comme document XML avec un codage littéral du document si un type de port WSDL est utilisé ou comme objet Java, si une interface Java est utilisée. La forme de données et de métadonnées préférée sont les objets de données, dans les services SCA (Service Component Architecture). Comme les composants, les objets métier séparent l'objet de données de son implémentation. Par exemple, un composant interagit avec les bons de commandes tandis que le bon de commande lui-même peut utiliser JDBC, EJB, etc. pour mettre à jour les données. Les objets métier laissent le développeur d'intégration se concentrer sur les artefacts métier. En fait, les objets de données de

service ne sont pas visibles par le développeur d'intégration. Ils sont définis par une demande de spécification Java (JSR) d'objets de données de service.

Dans la figure 15, des objets métier sont transmis d'un service externe à une exportation, d'une exportation à un composant, d'un composant à un composant, d'un composant à une importation et d'une importation à un service. Les importations et les exportations sont abordées dans la section Liaisons.

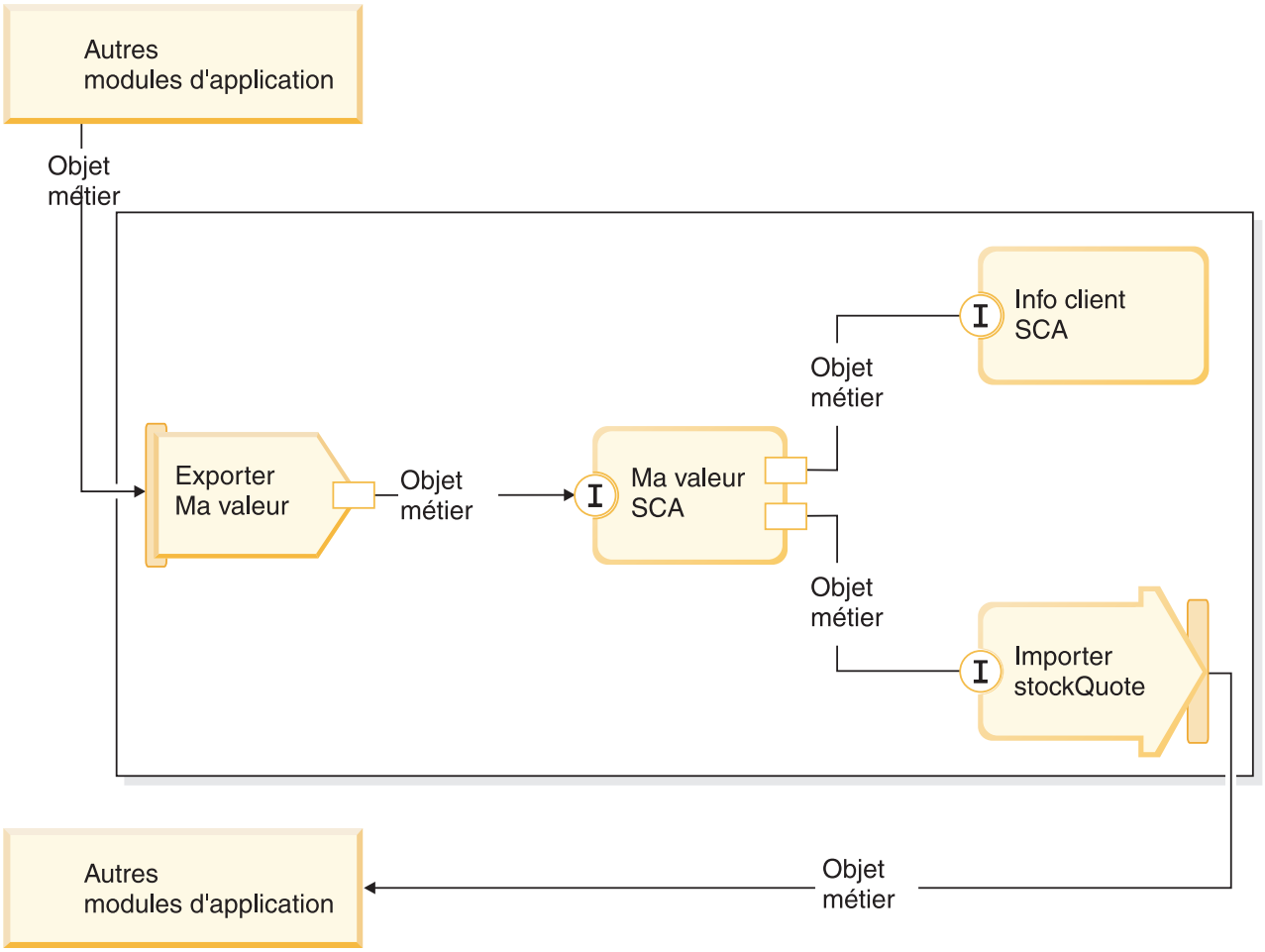


Figure 15. Objets métier

Objets métier

L'industrie du logiciel a développé plusieurs modèles et infrastructures de programmation dans lesquels des *objets métier* offrent une représentation naturelle des données de gestion dédiées au traitement des applications.

D'une manière générale, ces objets métier :

- Sont définis sur la base de normes de l'industrie
- 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

Vu sous l'aspect outils, WebSphere Integration Developer offre aux développeurs un modèle d'objet métier commun pour la représentation des divers types d'entités commerciales issues de domaines différents. Au moment du développement, ce modèle permet aux développeurs de définir des objets métier en tant que définitions de schémas XML.

Au moment de l'exécution, les données de gestion définies par les définitions de schémas XML sont représentées comme des 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.

Définir des objets métier

La définition d'objets métier s'effectue à l'aide de l'éditeur livré avec WebSphere Integration Developer. 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. WebSphere Integration Developer peut importer directement ces objets métier.

WebSphere Integration Developer 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, WebSphere Integration Developer autorise la création d'objets métier dans des bandothèques qui peuvent dès lors être associées à plusieurs modules d'application.

Les contrats pour les services fournis et utilisés par un module d'application SCA (Service Component Architecture), tout comme les contrats servant à créer les composants au sein d'un module d'application sont définis à l'aide du langage WSDL. Un WSDL peut représenter les opérations et les objets métier d'un contrat, les objets métier étant 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.

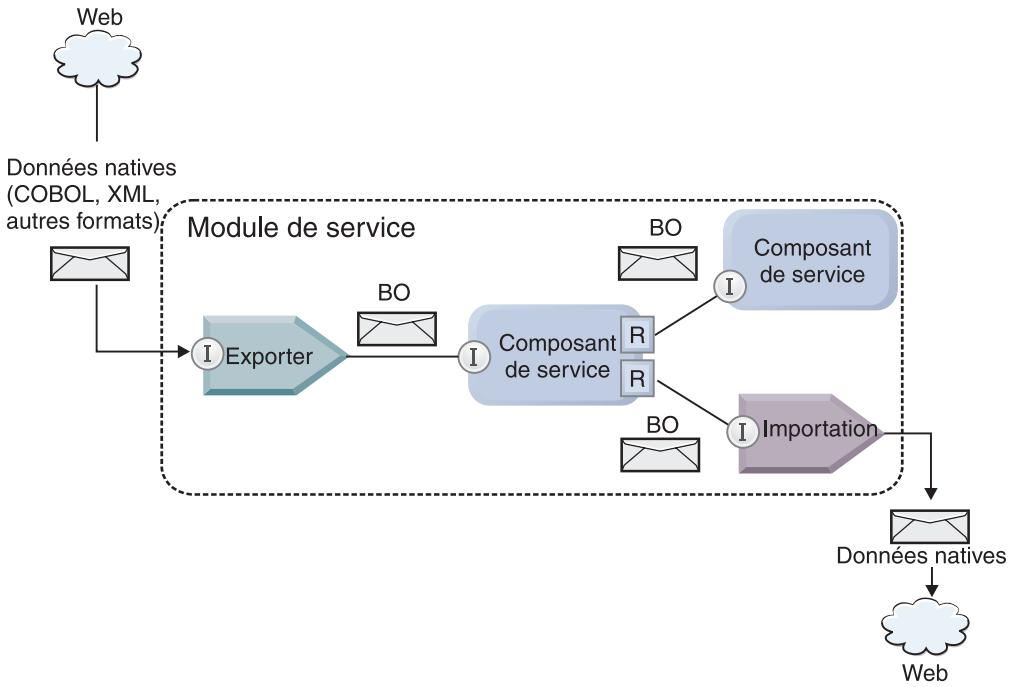


Figure 16. Les objets métier représentent les données qui circulent entre les services dans une application.

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://osoa.org/display/Main/Service+Data+Objects+Specifications.

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 et Package com.ibm.websphere.bo.

Liaisons, liaisons de données et gestionnaires de données

Comme le montre la figure 16, à la page 20, 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 et Liaisons de données.

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 ou du composant de flux de médiation et de sa primitive XML Map. 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 WebSphere Integration Developer, 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

WebSphere Integration Developer fournit une propriété pour les modules et les bibliothèques, qui permet de configurer le mode d'analyse du XML pour les objets métier : attentif ou passif.

- Si le mode doit être *attentif*, les flux des octets XML sont analysés attentivement pour la création de l'objet métier.
- Avec l'option *passif*, 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 attentive pour la création de l'objet métier.

Avantages comparés des deux modes d'analyse syntaxique : mode attentif et mode passif

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.

Nous allons exposer quelques conseils d'ordre général concernant les types d'applications qui bénéficient de chacun de ces deux modes :

- Applications tirant avantage du mode passif d'analyse syntaxique du XML

  Les applications qui analysent des flux importants de données XML constateront très probablement une amélioration de leurs performances si l'on utilise le mode passif. Ces améliorations se feront d'autant plus sentir qu'augmentera la taille des flux d'octets XML alors que la quantité des données analysées par l'application dans ce flux diminuera, quant à elle.

  Remarque : Le mode d'analyse syntaxique lente des objets métier est pris en charge dans WebSphere Process Server version 7.0.0.3 et versions ultérieures. Les modules et les modules de médiation incluant des composants de flux de médiation ne sont pas pris en charge.
- Applications tirant avantage du mode attentif d'analyse syntaxique du XML

  Les applications suivantes sont censées optimiser leurs performances dans ce mode :
  - les applications qui analysent des flux de données non XML
  - les applications qui sont créées à l'aide du service BOFactory
  - les applications qui analysent des messages XML de taille très modeste

Remarques concernant la migration et le développement d'applications

Si vous configurez une application initialement développée à l'aide du mode d'analyse syntaxique attentive pour utiliser dorénavant le mode d'analyse syntaxique lente, ou si vous prévoyez d'utiliser alternativement ces deux modes pour une application, tenez compte des différences existant entre ces deux modes ainsi que des remarques concernant 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 attentif, ces exceptions dès que l'objet métier est analysé dans le flux XML.
- Si c'est le mode passif 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 passif 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 passif ou s'il s'exécutait en mode attentif.

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 passif d'analyse du XML 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 passif intercepte des problèmes que ne voit pas le mode attentif et il fournit des messages d'erreur plus détaillés.

XML Maps version 602

Il est possible que les flux de médiation initialement développés avant WebSphere Integration Developer version 6.1 contiennent des primitives XSLT qui utilisent une mappe ou une feuille de style qui ne peut s'exécuter directement en mode d'analyse syntaxique XML lente. Lorsqu'on fait migrer une application pour l'utiliser en mode passif, les fichiers de mappes associés aux primitives XSLT peuvent être automatiquement mis à jour par l'assistant de migration pour qu'ils s'exécutent dans le nouveau mode. Cependant, si une primitive XSLT se réfère directement à une feuille de style ayant été éditée manuellement, cette dernière n'est pas migrée et ne peut pas être exécutée en mode d'analyse syntaxique lente.

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 attentif, 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 recommandé de supprimer les API privées de l'application.

API EMF d'objets de message de service

Un composant de médiation de WebSphere Process Server 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 passif, 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 passif, il est impossible de transtyper en objets EMF ServiceMessageObject et son contenu.

Service BOMode

Le service BOMode est utilisé pour déterminer si le mode d'analyse syntaxique XML en cours est attentif ou lent.

Migration

Toutes les applications antérieures à la version 7.0.0.0 s'exécutent en mode attentif. 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 attentif.

Pour permettre à une application antérieure à la version 7.0.0.0 d'utiliser le mode passif, vous devez préalablement utiliser WebSphere Integration Developer pour faire migrer les artefacts de l'application. Après la migration, vous pouvez alors configurer l'application pour qu'elle utilise le mode passif.

Pour plus d'informations sur la migration des artefacts dans WebSphere Integration Developer, voir Migrating source artifacts et pour plus d'informations sur la configuration du mode d'analyse syntaxique, voir Configuration du mode d'analyse syntaxique des objets métier des modules et bibliothèques.

Propriété d'objet métier de type QName

Vous devez modifier le code d'application à utiliser avec des objets métier contenant la propriété de type QName si vous souhaitez que votre application ayant utilisé l'analyse syntaxique attentive utilise l'analyse syntaxique lente. En mode d'analyse syntaxique attentive, WebSphere Process Server utilise la classe Java org.eclipse.emf.ecore.xml.type.internal.QName pour définir la valeur de propriété de type QName. Le mode d'analyse syntaxique lente utilise la classe Java javax.xml.namespace.QName pour définir la valeur de la propriété de type QName. Modifiez le code d'application lorsque vous passez un module du mode attentif au mode lent, en remplaçant la référence à la classe Java org.eclipse.emf.ecore.xml.type.internal.QName par la classe javax.xml.namespace.QName.

Relations

Une relation est une association entre deux ou plusieurs entités de données, généralement des objets métier. 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 WebSphere Process Server 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. Vous pouvez, par exemple, définir une relation *voiture-propriétaire*, sachant que le 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* ci-dessus, 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). Ces dernières utilisent chacune différents logiciels pour effectuer un suivi du personnel et des ordinateurs portables. Le groupe souhaite disposer d'un moyen d'effectuer un suivi des employés et de leurs ordinateurs portables et recherche une solution permettant :

- De visualiser tous les employés des différentes unités métier comme s'ils étaient regroupés dans une base de données unique
- D'avoir une vue centralisée de tous les ordinateurs portables
- De permettre aux employés de se connecter au système et d'acheter un ordinateur portable
- D'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 métier 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 patterns qui gèrent à votre place toutes ces instances de relations.

Outils de gestion des relations

L'*éditeur de relations* contenu dans WebSphere Integration Developer est un outil permettant de concevoir les rôles et les relations d'intégration métier. Pour obtenir des informations détaillées sur la création de relations et l'utilisation de l'éditeur de relations, consultez le centre de documentation de WebSphere Integration Developer.

Le *service de relation* est un service d'infrastructure de WebSphere Process Server 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 WebSphere Integration Developer. 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 WebSphere Process Server 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 WebSphere Process Server
- Transformations de relations dans le service de mappage d'objets métier de WebSphere Process Server
- Utilisation de l'outil de gestion de relations

Pour obtenir des informations détaillées sur la création de relations, l'identification de types de relations et l'utilisation de l'éditeur de relations, consultez le centre de documentation de WebSphere Integration Developer.

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 modèle de relation (dont correlate(), correlateforeignKeyLookup)
- Modèles de recherche de relation (API de recherche)

Le bus de service d'entreprise dans WebSphere Process Server

WebSphere Process Server prend en charge l'intégration des services d'application, et notamment 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 viaJava 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.

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 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.

Un bus de service d'entreprise, avec des médiations, 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 ainsi aux fonctions métier qu'ils représentent, sans avoir besoin que des programmeurs écrivent et entretiennent 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. Il permet d'intégrer des 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 d'échanger des informations dans des formats différents, le bus garantissant que l'information distribuée à une fonction métier est au format requis par cette 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.

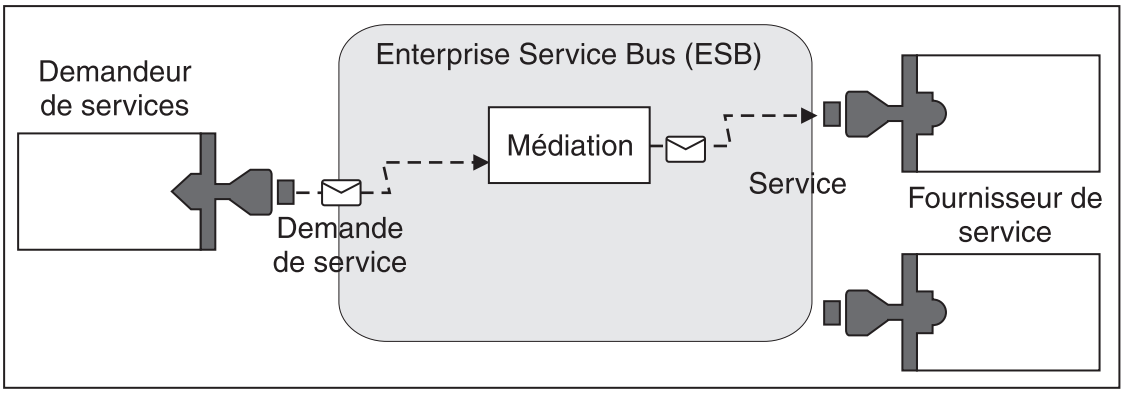


Figure 17. Bus de service d'entreprise. Le bus de service d'entreprise achemine les messages entre les applications, qui sont demandeurs ou fournisseurs de services. Le bus convertit les protocoles de transport ainsi que les formats des messages entre les demandeurs et les fournisseurs. Dans ce schéma, chaque application utilise un protocole différent (représenté par les différentes formes géométriques de leurs connecteurs) et utilise différents formats de message.

Grâce au bus de service d'entreprise, vous vous consacrez désormais entièrement à votre métier, sans vous soucier des systèmes informatiques. Vous pouvez apporter des modifications ou des ajouts aux services, au 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 du bus de services d'entreprise

WebSphere Process Server inclut des fonctions de bus de services d'entreprise. WebSphere Process Server prend en charge l'intégration de technologies orientées services, messages et événements afin d'offrir une infrastructure de messagerie normalisée dans un bus de service 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.

WebSphere Process Server est doté de la même technologie qu'IBM WebSphere Enterprise Service Bus. Cette technologie fait partie des fonctionnalités sous-jacentes de WebSphere Process Server 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 WebSphere Process Server. 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 WebSphere Process Server.

Vous pouvez déployer WebSphere Enterprise Service Bus dans votre entreprise, afin d'étendre la portée des connectivités offertes par les solutions d'intégration de processus motorisées par des installations distinctes de WebSphere Process Server ou d'autres solutions d'intégration dans le cadre d'un ESB fédéré. 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 WebSphere Process Server.

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, qui est un ensemble de tables dans une base de données ou un schéma qui stocke les données persistantes.

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 distincte. Vous pouvez aussi créer plusieurs magasins de données dans la même base de données, chacun 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.

Sources de données

Les sources de données établissent un lien entre les applications et les bases de données relationnelles.

Les applications utilisent une source de données pour se connecter à une base de données relationnelle. Une source de données est l'équivalent d'une connexion JCA (Java Connector Architecture (JCA), qui assure la connexion à d'autres types de systèmes d'information d'entreprise (EIS).

Une source de données est associée à un fournisseur JDBC, lequel procure les classes d'implémentation de pilotes requises par la connexion JDBC à un type spécifique de base de données. Les composants d'application procèdent à des transactions directement avec la source de données, pour obtenir des instances de connexion à la base de données. Chaque pool de connexions correspondant à chaque source de données assure la gestion des connexions.

Vous pouvez créer plusieurs sources de données avec des paramètres différents, et les associer au même fournisseur JDBC. Par exemple, vous pouvez utiliser plusieurs sources de données pour accéder à différentes bases de données appartenant à la même application de base de données. WebSphere Process Server exige que les fournisseurs JDBC implémentent l'une des interfaces de source de données suivantes, ou les deux, définies par Sun Microsystems. Ces interfaces permettent à l'application de s'exécuter selon un protocole de transaction à une ou deux phases.

Remarque : Les sources de données de Business Process Choreographer sont créées à l'aide des outils de configuration de Business Process Choreographer. Voir Configuration de Business Process Choreographer.

- ConnectionPoolDataSource : une source de données prenant en charge la participation de l'application aux transactions locales et globales, à l'exception des transactions de validation en deux phases. Lorsqu'une source de données de pool de connexion est impliquée dans une transaction globale, la récupération de la transaction n'est pas assurée par le gestionnaire de transactions. Si plusieurs gestionnaires de ressources sont impliqués, l'application est chargée du processus de récupération.
- XADataSource - une source de données prenant en charge la participation de l'application à n'importe quel environnement de transaction en une ou deux phases. Lorsque cette source de données est impliquée dans une transaction globale, le gestionnaire de transactions WebSphere Application Server prend en charge la récupération des transactions.

Les tableaux suivants fournissent des exemples de configurations d'environnements autonome et de déploiement :

Tableau 2. Configuration d'environnement autonome type

Source de données	Composant	Portée	Nom JNDI
Source de données de WBI	CommonDB	Noeud	jdbc/WPSDB
Source de données ME du bus d'application SCA	SCA ME	Serveur	jdbc/com.ibm.ws.sib/nlNode01.server1-SCA.APPLICATION.localhostNode01Cell.Bus

Tableau 2. Configuration d'environnement autonome type (suite)

Source de données	Composant	Portée	Nom JNDI
Source de données de Business Process Choreographer	BPC	Serveur	jdbc/BPEDB
Source de données ME de Business Process Choreographer	BPC ME	Serveur	jdbc/com.ibm.ws.sib/nlNode01.server1-BPC.localhostNode01Cell.Bus
événement	CEI	Serveur	jdbc/cei
Source de données ME de CEI	CEI ME	Serveur	jdbc/com.ibm.ws.sib/nlNode01.server1-CEI.*nomCellule*.BUS

Tableau 3. Configuration d'environnement de déploiement type

Source de données	Composant	Portée	Nom JNDI
Source de données de WBI	CommonDB	Cellule	jdbc/WPSDB
Source de données ME du bus d'application SCA	SCA ME	Cluster	jdbc/com.ibm.ws.sib/clusterone-SCA.APPLICATION.enduranceTestCell01.Bus
Source de données de Business Process Choreographer	BPC	Cluster	jdbc/BPEDB
Source de données ME de Business Process Choreographer	BPC ME	Cluster	jdbc/com.ibm.ws.sib/clusterone-BPC.enduranceTestCell01.Bus
événement	CEI	Cluster	jdbc/cei
Source de données ME de CEI	CEI ME	Cluster	jdbc/com.ibm.ws.sib/clusterone-CEI.*nomCellule*.BUS

Pour plus d'informations sur les sources de données, voir la rubrique «Sources de données» dans le centre de documentation de WebSphere Application Server.

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 du pilote JDBC requise pour accéder à une base de données fournisseur 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 WebSphere Process Server

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 WebSphere Process Server.

Certains bus sont automatiquement créés en vue d'être utilisés par le système, par les applications Service Component Architecture (SCA) que vous déployez et par d'autres composants. Vous pouvez également créer des bus pour la logique d'intégration de services ou pour d'autres applications (par exemple pour les applications faisant office de demandeurs et de fournisseurs de services dansWebSphere Process Server, ou pour établir une liaison avec WebSphere MQ.

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 bus de service d'entreprise.

Certains bus d'intégration de services sont créés automatiquement pour prendre en chargeWebSphere Process Server. Quatre bus maximum sont créés lorsque vous définissez un nouvel environnement de déploiement ou lorsque vous configurez un serveur ou un cluster afin de prendre en charge des applications SCA. Ces bus possèdent chacun trois alias d'authentification, que vous devez configurer.

Bus système SCA :

Le *bus système SCA* est un bus d'intégration de services utilisé pour héberger des destinations de files d'attente pour modules SCA (Service Component Architecture). L'environnement d'exécution SCA, qui prend en charge les modules de médiation, utilise des destinations de file d'attente sur le bus système comme une infrastructure pouvant prendre en charge les interactions asynchrones entre les composants et les modules.

Ce bus système est automatiquement créé lorsque vous créez un environnement de déploiement ou lorsque vous configurez un serveur ou un cluster afin de prendre en charge des applications SCA. Il offre une portée au sein de laquelle vous pouvez configurer des ressources, par exemple des destinations de files d'attente,

pour les modules de médiation et les points de contact d'interaction. Le bus permet l'acheminement de messages entre des points de contact. Vous pouvez indiquer la qualité de service associée au bus, notamment la priorité et la fiabilité.

Le nom du bus est SCA.SYSTEM.busID.Bus. L'alias d'authentification utilisé pour sécuriser ce bus est SCA_Auth_Alias.

Bus d'application SCA :

Les destinations du bus d'application prennent en charge la communication asynchrone des adaptateurs WebSphere Business Integration avec d'autres composants SOA.

Ce bus est automatiquement créé lorsque vous définissez un nouvel environnement de déploiement ou lorsque vous configurez un serveur ou un cluster afin de prendre en charge des applications SCA. Il est semblable aux bus d'intégration de services éventuellement créés pour prendre en charge une logique d'intégration de services ou d'autres applications.

Le nom du bus est SCA.APPLICATION.busID.Bus. L'alias d'authentification utilisé pour sécuriser ce bus est SCA_Auth_Alias.

Bus Common Event Infrastructure :

Le bus Common Event Infrastructure est utilisé pour la transmission asynchrone d'événements de base au serveur Common Event Infrastructure configuré.

Le nom de ce bus est CommonEventInfrastructure_Bus. L'alias d'authentification utilisé pour sécuriser ce bus est CommonEventInfrastructureJMSAuthAlias

Bus de Business Process Choreographer :

Utilisez le nom de bus et l'alias d'authentification du composant Business Process Choreographer pour la transmission interne des messages.

Le bus du composant Business Process Choreographer est utilisé pour la transmission interne des messages et pour l'API JMS (Java Messaging Service) de Business Flow Manager.

Le nom du bus est BPC.cellName.Bus. L'alias d'authentification est BPC_Auth_Alias

Applications et modules de service

Un module de service est un module SCA (Service Component Architecture) qui offre des services en phase d'exécution. Lorsque vous déployez un module de service sur WebSphere Process Server, 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 médiation permet aux applications de communiquer : il convertit

l'appel de service en un format compris par la cible, transmet la demande à la cible et renvoie le résultat à l'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 afficher les détails de l'application de service et de 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 détails du module de service que vous pouvez afficher dépendent du module SCA. Ils peuvent comprendre les attributs suivants.

- module SCAnom
- module SCAdescription
- Nom de l'application associée
- Informations de 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 par lequel un module SCA accède à un service. Par exemple, via SOAP/HTTP.
- Exportations module SCA :
  - Les interfaces d'exportation sont des définitions abstraites qui décrivent la façon dont des 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 module SCA et indirectement à un service.
- module SCAproperties

Importations et liaisons d'importation

Les importations définissent des interactions entre les modules SCA (Service Component Architecture) et les fournisseurs de services. Grâce aux importations, les modules SCA permettent aux composants d'accéder aux services externes (services qui se trouvent en dehors du module SCA) 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 module SCA requiert au moins une interface et une importation de module SCA est dotée d'une seule liaison.

- Les interfaces d'importation sont des définitions abrégées qui définissent un ensemble d'opérations via WSDL (Web Services Description Language), un langage XML utilisé pour décrire des services Web. Un module SCA peut disposer d'un grand nombre d'interfaces d'importation.
- 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

WebSphere Process Server 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 noeud final spécifié lors de l'importation. Ce noeud final est appelé noeud final 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 noeud final 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 noeud final. 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 de type de liaison supplémentaire 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 de type de liaison indiquant l'utilisation de SOAP 1.2 peut être spécifiée.

Exportation et liaisons d'exportation

Les exportations définissent des interactions entre les modules SCA (Service Component Architecture) et les demandeurs de services. Les exportations permettent aux modules SCA de proposer des services à d'autres modules. Les liaisons d'exportation définissent un mode d'accès spécifique du module SCA par des demandeurs de services.

Interfaces et liaisons

Une exportation de module SCA requiert au moins une interface.

- Les interfaces d'exportation sont des définitions abrégées qui définissent un ensemble d'opérations via WSDL (Web Services Description Language), un langage XML utilisé pour 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 seule liaison est spécifiée par exportation module SCA. Une opération d'exportation pour laquelle aucune liaison n'a été spécifiée, est interprétée comme une exportation dotée d'une liaison de type SCA lors de l'exécution.

Liaisons d'exportation prises en charge

Le WebSphere Process Server 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 service Web permettent d'appeler les exportations comme 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 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 (Service Component Architecture) 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. cela vous permet d'acheminer des messages à différents fournisseurs de services et de modifier le format ou le

contenu du message. Modules de médiation peuvent fournir des fonctions, telles que la consignation de message et le traitement des erreurs, qui sont adaptées à vos besoins.

Vous pouvez modifier certains aspects des modules de médiation à partir de la console d'administration WebSphere Process Server, 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 à partir de WebSphere Process Server 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 à partir de WebSphere Process Server.
- Composants SCA, blocs structurels des modules SCA tels que des modules de médiation. Vous pouvez créer et personnaliser des composants et des modules SCA graphiquement via WebSphere Integration Developer. Après avoir déployé un module de médiation, vous pouvez en personnaliser certains aspects à partir de la console d'administration WebSphere Process Server, 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. WebSphere Process Server 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'unmodule de médiation qui ne contient pas de composant de flux de médiation est de transformer 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 et apporter certaines modifications à des modules de médiation depuis WebSphere Process Server. Cependant, il n'est pas possible de visualiser ni de modifier des composants SCA à l'intérieur d'un module WebSphere Process Server. Utilisez WebSphere Integration Developer pour personnaliser les composants SCA.

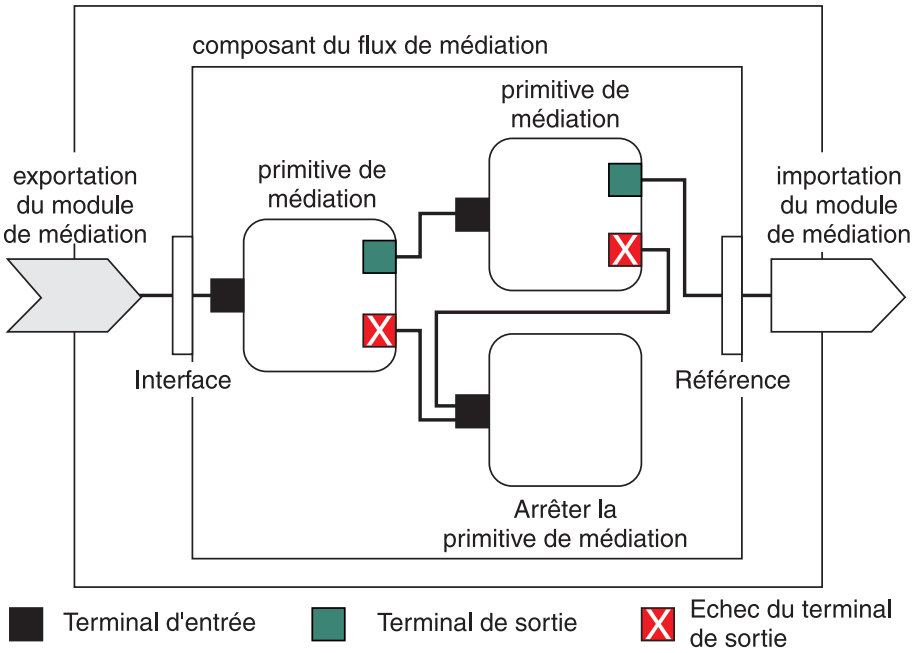


Figure 18. Exemple simplifié d'un module de médiation. Le module de médiation contient un composant de flux de médiation, qui contient des primitives de médiation.

- 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 depuis la console d'administration WebSphere Process Server, le développeur d'intégration doit promouvoir les propriétés. Certaines propriétés peuvent être configurées administrativement et WebSphere Integration Developer 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 due au fait que leur modification affecte le flux de médiation d'une manière qui nécessite le redéploiement du module de médiation. WebSphere Integration Developer 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 de WebSphere Process Server 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 WebSphere Integration Developer.
- Un module de médiation ou une bibliothèque dépendante peut également définir des sous-flux. Un flux secondaire encapsule un ensemble de primitives

de médiation reliées les unes aux autres sous la forme d'un élément de logique d'intégration réutilisable. 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 WebSphere Integration Developer, et généralement déployés sur WebSphere Process Server 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 WebSphere Integration Developer, puis ordonner à WebSphere Integration Developer de compiler le module de médiation dans un fichier JAR (Java archive), 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.

Logiquement, les Modules de médiation peuvent être considérés comme une entité. Toutefois, les modules SCA sont définis par un certain nombre de fichiers XML stockés dans un fichier JAR.

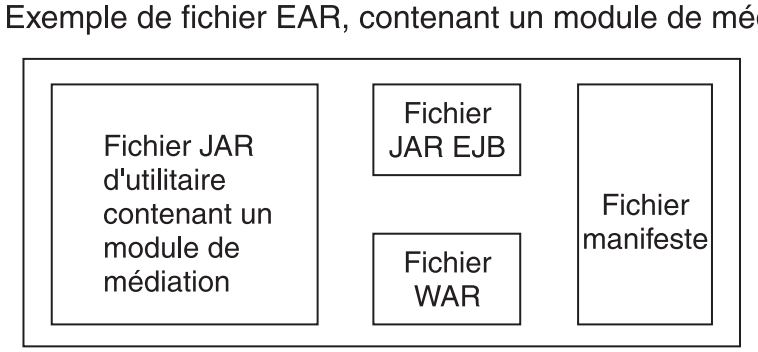


Figure 19. Exemple simplifié de fichier EAR contenant un module de médiation. Le fichier EAR contient des fichiers JAR. Le fichier JAR utilitaire contient un module de médiation.

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.

WebSphere Process Server prend en charge un ensemble intégré de primitives de médiation, qui mettent en oeuvre des fonctionnalités de médiation standard pour les modules de médiation ou modules déployés vers WebSphere Process Server. 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). Une

primitive de médiation peut également définir une opération «sortante» qui envoie des messages vers un autre composant ou module.

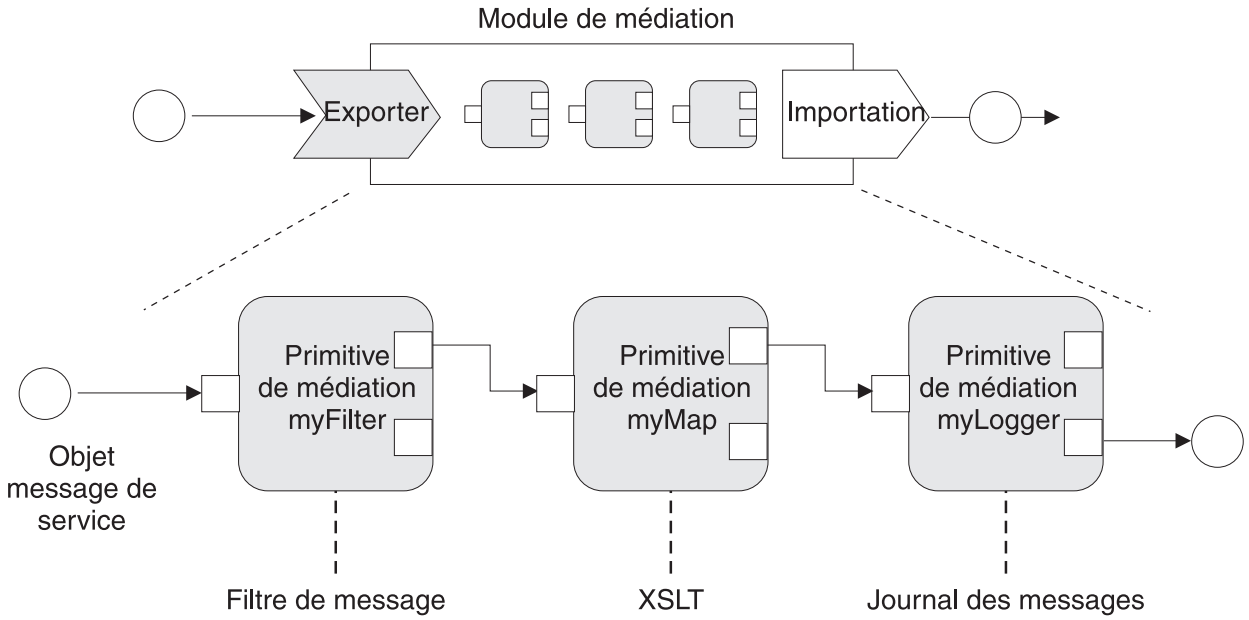


Figure 20. Module de médiation contentant trois primitives de médiation

Vous pouvez utiliser WebSphere Integration Developer 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.

WebSphere Integration Developer permet également de modéliser et assembler sous forme graphique les composants de flux de médiation à partir de primitives de médiation et d'assembler les modules de médiation ou modules à partir des composants de flux de médiation. La console d'administration fait référence aux modules de médiation et modules en tant que modules SCA.

WebSphere Integration Developer 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 WebSphere Process Server :

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 modèles 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 Invocation de service 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 de consultation de base de données. Utilisez la console d'administration pour vous simplifier la tâche.
- La primitive de médiation de consultation de base de données ne peut lire qu'une seule table.
- La colonne de clé spécifiée doit contenir une valeur unique.
- Les données dans la colonne des valeurs doivent être un type de schéma XML simple ou un type de schéma XML qui étend un type de schéma XML simple.

Consultation de noeud final (Endpoint Lookup)
Permet d'effectuer le routage dynamique de requêtes en recherchant les noeuds finaux 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.
- WebSphere Process Server doit savoir quel registre utiliser et, par conséquent, vous devez créer des définitions d'accès WSRR à l'aide de la console d'administration WebSphere Process Server.

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 les événements Emetteur d'événements via le navigateur CBE (Common Base Events) de WebSphere Process Server.
- 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 WebSphere Integration Developer génèrent un fichier de définitions pour le catalogue d'événements Common Event Infrastructure (CEI), pour chacune des primitives de médiation de l'émetteur d'événements configuré. Les fichier de définitions du catalogue d'événements sont des artefacts d'exportation destinés à vous venir en aide ; ils ne sont pas utilisés par WebSphere Integration Developer or ni par le programme d'exécution de WebSphere Process Server. 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 surveillance depuis WebSphere Process Server. Ainsi, vous pouvez surveiller les événements émis à partir d'importations et d'exportations.

Echec Arrête un chemin donné dans le flux, et génère une exception.

Fan In Permet de regrouper (d'associer) des messages.

- Ne peut être utilisé qu'en combinaison avec la primitive de médiation Sortance.
- L'association des primitives de médiation Fan Out et Fan In permet le regroupement de données dans un message de sortie.
- La primitive de médiation Entrance reçoit les messages jusqu'à ce qu'un point de décision soit atteint et un message est alors généré.
- Le contexte partagé permet de conserver les données de regroupement.

Fan Out

Permet de diviser et de regrouper (associer) des messages.

- L'association des primitives de médiation Fan Out et Fan In permet le regroupement de données dans un message de sortie.
- En mode d'itération, la primitive de médiation Sortance permet d'exécuter une itération dans un 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 les en-têtes de messages HTTP.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes HTTP.

Configurateur d'en-tête MQ

Fournit un mécanisme de gestion des en-têtes dans les messages MQ.

- Permet de créer, définir, copier ou supprimer des en-têtes de message MQ.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes MQ.

Configurateur d'en-tête SOAP

Fournit un mécanisme de gestion des en-têtes dans les messages SOAP.

- Peut créer, définir, copier ou supprimer les en-têtes de messages SOAP.
- Peut définir plusieurs actions pour modifier plusieurs en-têtes SOAP.

Définition d'éléments de messages

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 dans la colonne des valeurs doivent être un type de schéma XML simple ou un type de schéma XML qui étend 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 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 Journal des messages utilise la base de données Common. L'environnement d'exécution mappe la source de données à `jdbc/mediation/messageLog` sur 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.

Résolution de règle
Permet la configuration dyanamique 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 noeud final 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.
- WebSphere Process Server doit savoir quel registre utiliser et, par conséquent, vous devez créer des définitions d'accès WSRR à l'aide de la console d'administration WebSphere Process Server.

Invocation de service
Appelle un service depuis un flux de médiation au lieu d'attendre la fin du flux de médiation et d'utiliser le 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
Au cours du développement de l'intégration, permet de traiter les zones de messages faiblement typées comme zones fortement typées. Une zone est faiblement typée si elle peut contenir plusieurs types de données. Une zone est réputée fortement typée si son type et sa structure interne sont connus.

- En phase d'exécution, la primitive de médiation de définition du type de message (Set Message Type) vous permet de vérifier que le contenu d'un message correspond aux types de données attendus.

Stop Arrête un chemin donné dans le flux sans générer d'exception.

Filtre de type
Permet de diriger les messages vers un autre chemin d'un flux, en fonction de leur type.

Transformation XSL
Transforme les messages.

- Permet d'exécuter des transformations XSL (Extensible Stylesheet Language).
- Vous transformez les messages à l'aide de la transformation XSLT (Extensible Stylesheet Transformations) 1.0. Cette dernière agit sur une sérialisation XML du message.

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 concerne l'acheminement de message où le flux est dynamique et tous les points de contact possibles sont prédéfinis dans un module SCA (Service Component Architecture), et l'acheminement de message où le flux et la sélection du point de contact sont tous les deux dynamiques. Dans le dernier cas, les points de contact de service sont sélectionnés à partir d'une source externe, en phase d'exécution.

Sélection de noeud final dynamique

L'environnement d'exécution est doté d'une fonction de routage des messages de demande et de réponse vers une adresse de noeud final 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 noeud final 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 noeud final 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 noeud final 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 noeud final de service puis créer les modules SCA pour extraire les noeuds finaux du registre WSRR.

Lorsque vous développez des modules SCA, vous utilisez la primitive de médiation de 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 noeuds finaux, il doit alors utiliser une autre primitive de médiation pour sélectionner le noeud final à privilégier.

Contrôle des demandes de service à l'aide de règles de médiation

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 noeuds finaux 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.

Message Service Clients

Message Service Clients est disponible pour C/C++ et .NET pour permettre aux applications non Java de se connecter à bus de service d'entreprise.

Les Message Service Clients pour C/C++ et .NET proposent une API dénommée XMS qui offre le même ensemble d'interfaces que l'API JMS (Java Message Service). Message Service Client pour C/C++ comprend 2 implémentations de XMS, l'une pour les applications écrites en C et l'autre pour les applications écrites en C++. Message Service Client pour .NET comprend une implémentation complète de XMS, qui peut être utilisée par tout 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 de clients Java EE de WebSphere Application Server Network Deployment, y compris les clients Web Services, EJB et JMS.