Composants IBM WebSphere Business Integration
Adapter

Adapter for iSeries - Guide d'utilisation

Version 2.1.x

Composants IBM WebSphere Business Integration
Adapter

Adapter for iSeries - Guide d'utilisation

Version 2.1.x

Important

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant dans la section «Informations légales», à la page 83.

Remarque

Certaines captures d'écrans de ce manuel ne sont pas disponibles en français à la date d'impression.

13 septembre 2005

LE PRESENT DOCUMENT EST LIVRE ″EN L'ETAT″. IBM DECLINE TOUTE RESPONSABILITE, EXPRESSE OU IMPLICITE, RELATIVE AUX INFORMATIONS QUI Y SONT CONTENUES, Y COMPRIS EN CE QUI CONCERNE LES GARANTIES DE QUALITE MARCHANDE OU D'ADAPTATION A VOS BESOINS. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

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
Tour Descartes
92066 Paris-La Défense Cedex 50

Table des matières

Avis aux lecteurs canadiens v

A propos de ce document vii
Public visé vii
Documents connexes vii
Conventions typographiques viii

Nouveautés de la présente édition xi
Version 2.1.x xi
Version 2.0.x xi
Versions précédentes xi

Chapitre 1. Présentation 1
Présentation des systèmes iSeries et AS/400 1
Fonctionnement de l'adaptateur 3

Chapitre 2. Installation de l'adaptateur iSeries 5
Environnement de l'adaptateur pour iSeries 5
Installation de l'adaptateur iSeries et des fichiers associés 7
Structure de fichiers installée 7
Tâches à effectuer après l'installation 7

Chapitre 3. Configuration de l'adaptateur iSeries 9
Configuration du connecteur 9

Chapitre 4. Présentation des objets métier pour le connecteur. 29
Définition des métadonnées du connecteur 29
Présentation de la structure de l'objet métier 29
Structure des objets métier pour les programmes RPG, COBOL et Java 30
Structure Business Object pour les files d'attente de données iSeries 34
Configuration des méta-objets en vue de l'interrogation 34
Spécification de propriétés d'attribut d'objet métier 36
Spécification du texte de l'application au niveau de l'attribut de l'objet métier 37

Chapitre 5. Création et modification d'objets métier 39
Présentation de l'ODA pour iSeries 39
Génération de définitions d'objets métier 39
Indication des informations sur l'objet métier 45
Téléchargement d'objets métier 47

Chapitre 6. Identification et résolution des erreurs 49
Gestion des erreurs 49
Consignation 49
Messages de traçage 49

Annexe. Propriétés de configuration standard pour les connecteurs 53
Nouvelles propriétés 53
Présentation des propriétés de connecteur standard 53
Référence rapide des propriétés standard 55
Propriétés standard 62

Index 81

Informations légales 83
Informations sur les interfaces de programmation 85
Marques et marques de service 85

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.

A propos de ce document

La famille de produits IBM WebSphere Business Integration Adapters propose une connectivité d'intégration pour les technologies e-business de pointe, les applications d'entreprise, les systèmes propriétaires et les grands systèmes. Cette famille de produits propose des outils et des modèles destinés à la personnalisation, la création et la gestion des composants pour l'intégration des processus métier.

Ce document décrit l'installation, la configuration, le développement d'objets métier et la résolution des incidents de IBM WebSphere Business Integration Adapter for iSeries(TM).

Public visé

Ce document s'adresse aux consultants et utilisateurs de WebSphere qui mettent en oeuvre le connecteur dans le cadre d'un système d'intégration. Pour utiliser les informations de ce document, vous devez avoir des connaissances dans les domaines suivants :

- Développement du connecteur
- Développement de l'objet métier
- Architecture de l'application OS/400
- iSeries Integrated File System

Documents connexes

La documentation complète qui accompagne ce produit présente les caractéristiques et les fonctions communes à toutes les installations de composants WebSphere Business Integration Adapter, et inclut des supports de référence sur des composants spécifiques.

Vous pouvez télécharger la documentation associée sur les sites suivants :

Pour obtenir des informations générales sur un adaptateur, pour apprendre à utiliser des adaptateurs avec des courtiers de messages WebSphere (WebSphere MQ Integrator, WebSphere MQ Integrator Broker, WebSphere Business Integration Message Broker) et à utiliser des adaptateurs avec WebSphere Application Server, voir le site Web :

http://www.ibm.com/websphere/integration/wbiadapters/infocenter

Pour utiliser des adaptateurs avec InterChange Server:

http://www.ibm.com/websphere/integration/wicserver/infocenter

Pour plus d'informations sur les courtiers de messages (WebSphere MQ Integrator Broker, WebSphere MQ Integrator et WebSphere Business Integration Message Broker) :

http://www.ibm.com/software/integration/mqfamily/library/manualsa/

Pour plus d'informations sur WebSphere Application Server :

http://www.ibm.com/software/webservers/appserv/library.html

Ces sites contiennent des explications simples pour télécharger, installer et afficher la documentation.

Pour plus d'informations sur JT400 et iSeries Access, voir :

http://publib.boulder.ibm.com/pubs/html/as400/infocenter.html

Remarque : Des informations importantes relatives à ce produit peuvent être disponibles dans les flashes de support technique (Technical Support Flashes), après la publication de ce document. Pour les consulter, accédez au site du support de WebSphere Business Integration,

http://www.ibm.com/software/integration/websphere/support/

Sélectionnez la zone qui vous intéresse et parcourez les sections Technotes et Flashes.

Conventions typographiques

Ce document utilise les conventions suivantes :

`Police courier`	Indique une valeur littérale, comme le nom d'une commande, le nom d'un fichier, des informations que vous tapez ou que le système affiche à l'écran.
gras	Indique un nouveau terme à sa première occurrence.
italique `italic`	Indique un nom de variable ou une référence croisée.
souligné en bleu	Le soulignement en bleu, visible uniquement lorsque vous consultez le document en ligne, indique un hyperlien de référence croisée. Si vous cliquez sur le terme souligné, vous êtes renvoyé à l'objet de la référence.
`{ }`	Dans une ligne de syntaxe, les accolades entourent un ensemble d'options parmi lesquelles vous ne devez sélectionner qu'une seule.
`[ ]`	Dans une ligne de syntaxe, les crochets entourent un paramètre facultatif.
`...`	Dans une ligne de syntaxe, les points de suspension indiquent une répétition du paramètre précédent. Par exemple, `option[,...]` signifie que vous pouvez entrer plusieurs options séparées par des virgules.
`< >`	Dans une convention de dénomination, les signes inférieur à et supérieur à entourent les différents éléments d'un nom afin de pouvoir les différencier les uns des autres, par exemple, `<nom_serveur><`*`nom_connecteur`*`>tmp.log`.
`/, \`	Dans ce document, les barres obliques inverses (\) sont utilisées comme convention pour les chemins de répertoire. Pour les systèmes UNIX, remplacez les barres obliques inverses par des barres obliques (/). Tous les noms de chemin des produits de systèmes d'intégration WebSphere sont relatifs au répertoire dans lequel le produit est installé sur votre système.

%texte% et *$texte*	Du texte placé entre des signes pourcentage (%) indique la valeur de la variable système `texte` Windows ou la variable utilisateur. Le symbole équivalent dans un environnement UNIX est *$texte*, indiquant la valeur de la variable d'environnement UNIX *texte*.
ProductDir	Correspond au répertoire où le produit est installé.

Nouveautés de la présente édition

Version 2.1.x

- iSeries ODA prend en charge les programmes source RPG et RPGLE
- Prise en charge de AIX 5.3, Linux RedHat 3.0 Update 1, Linux SuSe 8.1 SP3 et Java Toolbox V5.3.0.6. Pour plus d'informations sur la prise en charge des matériels et logiciels, voir «Environnement de l'adaptateur pour iSeries», à la page 5.
- Prise en charge de IBM Tivoli License Manager (ITLM)
- L'adaptateur peut désormais appeler des fichiers .PGM écrits en COBOL ou JAVA. Le nouveau verbe ″CALLPGM″ a été ajouté pour effectuer cette fonction.
- Prise en charge de JVM 1.4.2
- Prise en charge de la globalisation pour l'adaptateur iSeries et l'ODA. L'adaptateur et l'ODA prendront en charge les langues G1, mais pas l'ASI.

Version 2.0.x

- Prise en charge de l'accès aux files d'attente de données sur iSeries
- Prise en charge de l'interrogation pour surveiller les files d'attente de données. L'adaptateur est à présent bidirectionnel, c'est-à-dire qu'il prend en charge le traitement des événements et des requêtes
- Fonction ODA de génération de spécifications BO depuis les programmes sources RPG et les files d'attente de données, sur les machines iSeries
- Fonction permettant d'appeler plusieurs fois un PGM RPG à l'aide d'une même requête BO. La BO peut désormais retourner les valeurs de plusieurs appels sur le même programme RPG
- Un problème concernant le fait que le PGM RPG était appelé deux fois dans un seul appel a également été résolu dans cette version.

Versions précédentes

Les modifications apportées aux versions précédentes sont décrites dans les sections ci-dessous.

Version 1.1.x

Ajout de la prise en charge de Report Program Generator (RPG).

A partir de la version 1.1, l'adaptateur pour iSeries n'est plus pris en charge sous Microsoft Windows NT.

Les informations relatives à l'installation de l'adaptateur ont été déplacées. Voir le chapitre 2, «Installation de l'adaptateur iSeries et des fichiers associés», à la page 7, pour savoir où sont ces informations.

Version 1.0.x

La Version 1.0.x est la première édition de IBM WebSphere Business Integration Adapter for iSeries. L'adaptateur permet d'exécuter des programmes Report Program Generator (RPG) 4 sur un système iSeries ou AS/400.

Chapitre 1. Présentation

Le présent chapitre décrit IBM WebSphere Business Integration Adapter for iSeries. Grâce à IBM Toolbox for Java (un ensemble de classes Java(TM)), l'adaptateur permet d'exécuter des programmes RPG et d'accéder à des files d'attente de données sur un système iSeries ou AS/400. IBM Toolbox for Java fournit un ensemble de classes permettant d'accéder à des files d'attente de données et d'exécuter des programmes. L'adaptateur utilise ces classes et informations à partir de l'objet métier entrant, pour définir une liste de paramètres permettant l'exécution du programme et la lecture/écriture depuis/vers des files d'attente de données. L'adaptateur iSeries prend actuellement en charge le traitement des requêtes et des événements.

Les adaptateurs sont constitués d'un composant spécifique à l'application et de l'architecture de connecteur. Le composant propre à l'application contient des codes adaptés à une application ou une technologie spécifique. L'architecture de connecteur, dont le code est commun à tous les adaptateurs, joue le rôle d'intermédiaire entre le courtier d'intégration et le composant propre à l'application. L'architecture de connecteur fournit les services suivants entre le courtier d'intégration et le composant propre à l'application :

- Elle envoie les objets métier
- Elle assure l'échange des messages de démarrage et d'administration.

Ce document contient des informations sur le composant spécifique à l'application et sur l'architecture de connecteur. Il fait référence à ces deux éléments comme étant le connecteur.

Pour plus d'informations sur la relation qui existe entre le courtier d'intégration et l'adaptateur, voir le *IBM WebSphere InterChange Server System Administration Guide*, ou le *IBM WebSphere Business Integration Implementation Guide for WebSphere MQ Integrator Broker*.

Ce chapitre contient les sections suivantes :

- «Présentation des systèmes iSeries et AS/400»
- «Fonctionnement de l'adaptateur», à la page 3

Présentation des systèmes iSeries et AS/400

Le système IBM iSeries (appelé AS/400) est une plateforme de serveur fiable et hautement intégrée, qui permet aux entreprises d'exécuter simultanément plusieurs systèmes d'exploitation. Grâce à ses caractéristiques d'intégrité et de sécurité, il est utilisé dans de nombreuses applications critiques.

RPG a évolué d'un simple Report Program Generator (d'où il tire son nom) à un puissant langage procédural de développement d'applications sur machines iSeries. Il est actuellement pris en charge en environnement ILE (Integrated Language Environment) sur le iSeries.

Les serveurs hôtes traitent les requêtes à partir des PC clients ou d'autres appareils exécutant une application, tel qu'illustré dans la figure 1, à la page 2, pour permettre d'imprimer un document ou de réaliser d'autres tâches. Les ordinateurs iSeries et AS/400 sont des serveurs toutes fonctions, capables d'exécuter de

nombreuses tâches simultanément, telles que des opérations de fichiers, bases de données, applications, messagerie, impression, multimédia, fax et communications sans fil. Chaque serveur de tâches s'exécute en tant que travail séparé sur le système, et chaque travail de serveur envoie et reçoit des flux de données sur une connexion socket.

L'un de ces serveurs hôtes est le serveur d'appel de programme Remote Command and Distributed. Ce serveur exécute les programmes sur un système iSeries ou AS/400.

IBM Toolbox for JAVA intègre de nombreux modules capables de prendre en charge des fonctionnalités différentes. Par exemple, les classes Access gèrent les informations de connexion, créent et maintiennent les connexions socket, envoient et reçoivent les données, tandis que les classes Command Call exécutent des commandes de traitement par lot iSeries et AS/400.

L'adaptateur iSeries d'IBM utilise les classes Access et Program Call pour appeler le programme RPG. Les classes de conversion de données permettent de convertir les données alphanumériques entre les formats iSeries ou AS/400 et Java.

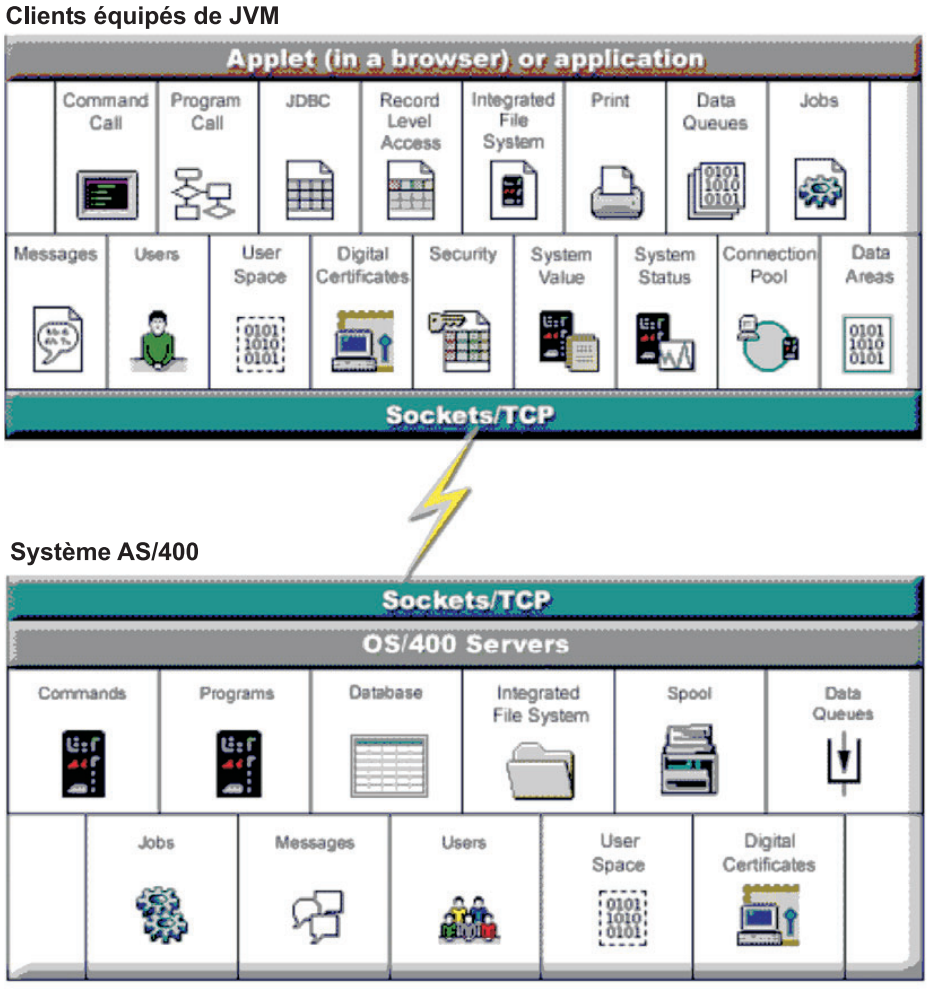


Figure 1. Présentation de l'architecture client-serveur AS/400

Alors que OS/400 exécuté sur un AS/400 est capable de gérer de nombreux types de tâches, l'adaptateur iSeries n'utilise que le serveur d'appel de programme Remote Command and Distributed. Ce serveur exécute les programmes sur le système AS/400.

La figure 2, à la page 3 montre un diagramme de la connexion de l'adaptateur iSeries du client au serveur.

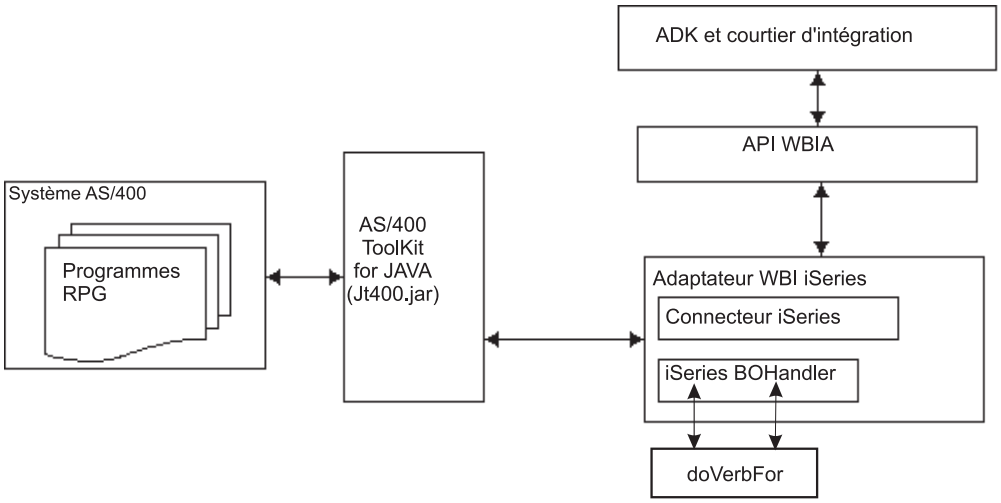


Figure 2. Diagramme des connexions de l'adaptateur iSeries

Files d'attente de données

Les files d'attente de données du iSeries permettent aux travaux de communiquer rapidement. Elles constituent par conséquent un excellent moyen pour synchroniser et transmettre les données entre les travaux. Grâces aux files d'attente de données du iSeries :

- De nombreux travaux peuvent accéder simultanément aux files d'attente de données
- Les messages d'une file d'attente de données sont dans un format libre
- La file d'attente de données peut servir pour les traitement synchrones et asynchrones
- Les messages d'une file d'attente de données peuvent être organisés de l'une des façons suivantes :
  - Dernier rentré, premier sorti (LIFO)
  - Premier rentré, premier sorti (FIFO)
  - Par clé

Chaque message d'une file d'attente de données indexées a une clé. Un message peut être extrait de la file d'attente simplement en précisant la clé associée.

Fonctionnement de l'adaptateur

Les sections qui suivent décrivent comment l'adaptateur traite les objets métier.

Traitement des objets métier

L'adaptateur reçoit les requêtes d'objet métier depuis un courtier d'intégration et détermine la liste des paramètres pour le programme RPG. Il établit ensuite la connexion au système iSeries ou AS/400 et exécute le programme.

L'objet métier entrant contient l'attribut enfant de connexion. Les informations de cet attribut sont utilisées pour la connexion au système iSeries ou AS/400.

Opérations du connecteur

Le connecteur transmet les objets métier entre le courtier d'intégration et un gestionnaire d'objets métier. Le connecteur :

1. Enregistre le BOHandler sur l'architecture.
2. L'architecture envoie la requête BO au BOHandler.
3. Le BOHandler utilise les informations sur l'attribut contenues dans l'objet métier entrant pour établir la liste des paramètres du programme RPG.
4. Le BOHandler appelle le programme RPG exécuté sur le système iSeries ou AS/400.

   Remarque : Il s'agit essentiellement d'un appel à exécuter un programme RPG sur le système iSeries ou AS/400, qui retourne ensuite une message de réussite ou d'erreur.
5. Le BOHandler retourne ensuite les résultats de l'exécution du programme RPG à l'architecture de l'adaptateur. Il renseigne également l'objet métier avec les paramètres retournés.

L'adaptateur est écrit en Java et composé de deux éléments :

- Le connecteur
- Le gestionnaire d'objet métier

Chapitre 2. Installation de l'adaptateur iSeries

Le présent chapitre décrit le processus d'installation et de configuration du connecteur. Il contient les sections suivantes :

- «Environnement de l'adaptateur pour iSeries»
- «Installation de l'adaptateur iSeries et des fichiers associés», à la page 7
- «Structure de fichiers installée», à la page 7
- «Tâches à effectuer après l'installation», à la page 7

Environnement de l'adaptateur pour iSeries

Avant d'installer, de configurer et d'utiliser l'adaptateur, vous devez connaître les caractéristiques nécessaires de son environnement. En complément des sections ci-dessous, les caractéristiques matérielles et logicielles requises sont indiquées dans le document technique suivant :

http://www.ibm.com/support/docview.wss?uid=swg27006249

- Compatibilité du courtier
- Conditions logicielles requises
- Plateformes de l'adaptateur

Compatibilité du courtier

L'architecture qu'utilise l'adaptateur doit être compatible avec la version du courtier d'intégration avec lequel l'adaptateur communique. La version 2.0 de l'adaptateur pour iSeries est prise en charge sur les versions de l'architecture de l'adaptateur et les courtiers d'intégration suivants :

Architecture de l'adaptateur : WebSphere Business Integration Adapter Framework, versions 2.6 et 2.7.

Courtiers d'intégration :

- WebSphere InterChange Server, version 4.2.2, 4.3
- WebSphere MQ Integrator, version 2.1.0
- WebSphere MQ Integrator Broker, version 2.1.0
- WebSphere Business Integration Message Broker, version 5.0
- WebSphere Application Server Enterprise, version 5.0.2, avec WebSphere Studio Application Developer Integration Edition, version 5.0.1
- WebSphere Business Integration Server Foundation 5.1, 5.1.1

Voir les Notes d'édition pour connaître les exceptions.

Remarque : Pour plus d'informations sur l'installation du courtier d'intégration et les composants requis, voir les documents suivants. Pour WebSphere InterChange Server (ICS), voir *System Installation Guide for UNIX* ou *for Windows*.

Pour les courtiers de messages (WebSphere MQ Integrator Broker, WebSphere MQ Integrator et WebSphere Business Integration Message Broker), voir *Implementing Adapters with WebSphere Message Brokers*, et

la documentation d'installation du courtier de messages. Certaines de ces informations sont disponibles à l'adresse http://www.ibm.com/software/integration/mqfamily/library/manualsa/.

Pour WebSphere Application Server, voir *Implementing Adapters with WebSphere Application Server* et la documentation disponible à l'adresse http://www.ibm.com/software/webservers/appserv/library.html.

Platesformes d'adaptateur

L'adaptateur fonctionne sur les platesformes suivantes :

- Windows 2000, 2003
- Sun Solaris 8, 9
- AIX 5.2, 5.3
- HP-UX 11i
- Linux RedHat AS/ ES/WS 3.0 avec Update 1
- Linux SuSe 8.1 avec SP3

Conditions requises

Pour utiliser le connecteur, votre environnement doit disposer de :

1. Fichiers Java et jar :
   - JDK 1.3 ou supérieur
   - Java Secure Socket Extension 1.0 (JSSE)
   - Fichier Jt400.jar

     Remarque : Le fichier IBM Toolbox for Java (licence produit 5722-JC1) V5R2 téléchargeable sur le site Web Toolbox à l'adresse : http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm.
     Le fichier jt400.jar doit être copié dans le répertoire iSeries %Product_dir%\connectors\ .
   - fichier WBIA.jar
   - fichier CrossWorlds.jar
   - fichier BIA_iSeries.jar
2. L'adaptateur iSeries est conçu pour se connecter à un AS/400 doté d'une des versions OS/400 suivantes :
   - Version 5, Edition 1
   - Version 4, Editions 1 à 3
3. L'option Host Servers de OS/400 doit être installée et en cours de fonctionnement.

   Remarque : Le serveur de files d'attente de données OS/400 exige des PTF pour exécuter correctement les fonctions peek. Vous devez vous procurer le PTF approprié à l'adresse :

   ```
   http://www-1.ibm.com/servers/eserver/iseries/toolbox/
   hostservicepackdetail.htm
   ```
4. Vous devez exécuter RPG III ou IV.

Installation de l'adaptateur iSeries et des fichiers associés

Pour plus d'informations sur l'installation des produits de WebSphere Business Integration, voir le guide *Installing WebSphere Business Integration Adapters* dans l'Infocenter de WebSphere Business Integration Adapters, sur le site Web suivant :

http://www.ibm.com/websphere/integration/wbiadapters/library/infocenter

Structure de fichiers installée

L'installation de l'adaptateur copie sur votre système les fichiers standard associés au connecteur. L'utilitaire installe le connecteur dans le répertoire *ProductDir*\connectors\iSeries et ajoute un raccourci pour le connecteur dans le menu de démarrage.

Remarque : *ProductDir* correspond au répertoire dans lequel le produit est installé.

Le Tableau 1 décrit la structure de fichiers utilisée par le connecteur et indique les fichiers qui sont automatiquement installés lorsque vous installez le connecteur à l'aide du programme d'installation.

Tableau 1. Structure de fichier du connecteur

Sous-répertoire de *ProductDir*	**Description**
\connectors\iSeries\BIA_iSeries.jar	Contient uniquement les classes utilisées par le connecteur iSeries
\connectors\iSeries\start_iSeries.bat	Script de démarrage du connecteur iSeries (Windows)
\connectors\iSeries\start_iSeries.sh	Script de démarrage du connecteur iSeries (Unix)
\connectors\iSeries\ext\	Répertoire dans lequel les fichiers .jar générés par ODA peuvent être enregistrés. Si vous les enregistrez dans ce répertoire, indiquez-le dans le script de démarrage (start_iSeries.bat ou start_iSeries.sh).
\connectors\messages\ BIA_iSeriesAdapter.txt	Fichier de message du connecteur
\ODA\iSeries\BIA_iSeriesODA.jar	ODA iSeries
\ODA\iSeries\start_iSeriesODA.bat	Fichier de démarrage de l'ODA (Windows)
\ODA\iSeries\start_iSeriesODA.sh	Fichier de démarrage de l'ODA (Unix)
Data\App\BIA_iSeriesAdapterTemplate	Définition du référentiel du connecteur.

Tâches à effectuer après l'installation

Après l'installation et avant le démarrage, vous devez configurer l'adaptateur. Pour plus d'informations, voir le Chapitre 3, «Configuration de l'adaptateur iSeries», à la page 9.

Chapitre 3. Configuration de l'adaptateur iSeries

Le présent chapitre décrit la configuration du connecteur. Il contient les sections suivantes :

- «Configuration du connecteur»

Configuration du connecteur

L'adaptateur iSeries utilise des propriétés standard de configuration et des propriétés spécifiques, indiquées dans les sections qui suivent.

Cette section aborde les points suivants :

- «Présentation de Connector Configurator»
- «Démarrage de Connector Configurator», à la page 10
- «Exécution de Configurator à partir de System Manager», à la page 11
- «Création d'un modèle de propriété spécifique au connecteur», à la page 11
- «Création d'un nouveau fichier de configuration», à la page 14
- «Définition des propriétés du fichier de configuration», à la page 16
- «Enregistrement du fichier de configuration», à la page 24
- «Modification d'un fichier de configuration», à la page 24
- «Exécution de la configuration», à la page 24
- «Utilisation de Connector Configurator en environnement globalisé», à la page 25
- «Démarrage du connecteur», à la page 25
- «Arrêt du connecteur», à la page 27
- «Création de plusieurs instances de connecteur», à la page 27

Présentation de Connector Configurator

Connector Configurator vous permet de configurer le connecteur de votre adaptateur à utiliser avec ces courtiers d'intégration :

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator, WebSphere MQ Integrator Broker et WebSphere Business Integration Message Broker, appelés courtiers de messages WebSphere (WMQI) ;
- WebSphere Application Server (WAS)

Connector Configurator vous permet de :

- créer un **modèle de propriété spécifique au connecteur** pour la configuration de votre connecteur ;
- créer un **fichier de configuration du connecteur** (vous devez créer un fichier de configuration pour chaque connecteur installé) ;
- définir les propriétés dans un fichier de configuration.
  Vous devez peut-être modifier les valeurs par défaut définies pour les propriétés dans les modèles du connecteur. Vous devez également déterminer les définitions d'objet métier prises en charge, indiquer les mappes à utiliser avec

les collaborations à l'aide d'ICS et spécifier les paramètres d'application de messagerie, de journalisation, de trace ainsi que ceux du gestionnaire de données, le cas échéant.

Le mode dans lequel vous exécutez Connector Configurator et le type de fichier de configuration que vous utilisez peuvent différer en fonction du courtier d'intégration que vous exécutez. Par exemple, si vous utilisez WMQI comme courtier, vous exécutez Connector Configurator directement, et non à partir de System Manager (voir «Exécution de Configurator en mode autonome»).

Les propriétés de configuration du connecteur incluent des propriétés de configuration standard (les propriétés communes à tous les connecteurs) et des propriétés spécifiques au connecteur (propriétés requises par le connecteur pour une technologie ou une application spécifique).

Dans la mesure où les **propriétés standard** sont utilisées par tous les connecteurs, vous n'avez pas besoin de définir ces propriétés de tout pièce ; Connector Configurator les incorpore à votre fichier de configuration dès que vous créez ce fichier. Cependant, vous devez définir la valeur de chaque propriété standard dans Connector Configurator.

L'intervalle des propriétés standard peut être différent pour tous les courtiers et toutes les configurations. Certaines propriétés ne sont disponibles que si vous attribuez une valeur spécifique à d'autres propriétés. La fenêtre des propriétés standard dans Connector Configurator affiche les propriétés disponibles pour votre configuration spécifique.

Cependant, pour les **propriétés spécifiques au connecteur**, vous devez d'abord définir les propriétés, puis leur attribuer une valeur. Pour ce faire, créez un modèle de propriétés spécifiques au connecteur pour votre adaptateur particulier. Il se peut qu'un modèle soit déjà configuré dans votre système, auquel cas vous pouvez l'utiliser. Dans le cas contraire, suivez les étapes dans la section «Création d'un modèle», à la page 11 pour configurer un nouveau modèle.

Remarque : Connector Configurator s'exécute uniquement dans un environnement Windows. Si vous exécutez le connecteur dans un environnement UNIX, utilisez Connector Configurator dans Windows pour modifier le fichier de configuration, puis copiez le fichier dans votre environnement UNIX.

Démarrage de Connector Configurator

Vous pouvez démarrer et exécuter Connector Configurator dans l'un de ces deux modes :

- de manière indépendante, en mode autonome ;
- à partir de System Manager.

Exécution de Configurator en mode autonome

Vous pouvez exécuter Connector Configurator en mode autonome et utiliser les fichiers de configuration indépendamment de votre courtier.

Pour ce faire, procédez comme suit :

- Dans **Démarrer>Programmes**, cliquez sur **IBM WebSphere InterChange Server>IBM WebSphere Business Integration Tools>Connector Configurator**.
- Sélectionnez **File>New>Connector Configuration**.

- Lorsque vous cliquez sur le menu déroulant en regard de **System Connectivity Integration Broker**, vous pouvez sélectionner ICS, WebSphere Message Brokers ou WAS, en fonction de votre courtier.

Vous pouvez choisir d'exécuter Connector Configurator en mode autonome pour créer le fichier, puis de vous connecter à System Manager afin de l'enregistrer dans un projet System Manager (voir «Remplissage d'un fichier de configuration», à la page 16).

Exécution de Configurator à partir de System Manager

Vous pouvez exécuter Connector Configurator à partir de System Manager.

Pour exécuter Connector Configurator, procédez comme suit :

1. Ouvrez System Manager.
2. Dans la fenêtre System Manager, développez l'icône **Integration Component Libraries** et mettez en évidence **Connecteurs**.
3. Dans la barre de menus de System Manager, cliquez sur **Outils>Connector Configurator**. La fenêtre Connector Configurator affiche la boîte de dialogue **New Connector**.
4. Lorsque vous cliquez sur le menu déroulant en regard de **System Connectivity Integration Broker**, vous pouvez sélectionner ICS, WebSphere Message Brokers ou WAS, en fonction de votre courtier.

Pour modifier un fichier de configuration existant, procédez comme suit :

- Dans la fenêtre System Manager, sélectionnez l'un des fichiers de configuration répertoriés dans le dossier du connecteur et cliquez dessus avec le bouton droit. Connector Configurator affiche le fichier de configuration avec le type de courtier d'intégration et le nom de fichier dans la partie supérieure.
- Dans Connector Configurator, sélectionnez **File>Open**. Sélectionnez le nom du fichier de configuration du connecteur dans un projet ou dans le répertoire dans lequel il est stocké.
- Cliquez sur l'onglet Standard Properties pour afficher les propriétés contenues dans ce fichier de configuration.

Création d'un modèle de propriété spécifique au connecteur

Pour créer un fichier de configuration pour votre connecteur, vous avez besoin d'un modèle de propriétés spécifiques au connecteur et des propriétés standard fournies par le système.

Vous pouvez créer un nouveau modèle pour les propriétés spécifiques au connecteur ou utiliser comme modèle une définition de connecteur existante.

- Pour créer un nouveau modèle, voir «Création d'un modèle».
- Pour utiliser un fichier existant, il vous suffit de modifier un modèle existant et de l'enregistrer sous le nouveau nom. Vous pouvez trouver des modèles existants dans le répertoire `\WebSphereAdapters\bin\Data\App`.

Création d'un modèle

Cette section décrit comment créer des propriétés dans le modèle, définir les valeurs et les caractéristiques générales de ces propriétés et indiquer toutes les dépendances entre les propriétés. Ensuite, vous pouvez utiliser le modèle comme base pour la création d'un nouveau fichier de configuration du connecteur.

Pour créer un modèle dans Connector Configurator, procédez comme suit :

1. Cliquez sur **File>New>Connector-Specific Property Template**.
2. La boîte de dialogue **Connector-Specific Property Template** s'affiche.
   - Entrez le nom du nouveau modèle dans la zone **Name** située sous **Input a New Template Name**. Vous voyez de nouveau ce nom lorsque vous ouvrez la boîte de dialogue pour créer un fichier de configuration à partir d'un modèle.
   - Pour afficher les définitions de propriétés spécifiques au connecteur dans n'importe quel modèle, sélectionnez le nom de ce modèle dans l'écran **Template Name**. La liste des définitions de propriétés contenues dans ce modèle apparaît dans l'écran **Template Preview**.
3. Vous pouvez utiliser un modèle existant dont les définitions de propriétés sont similaires à celles requises par votre connecteur comme point de départ pour votre modèle. Si aucun des modèles n'affiche les propriétés spécifiques au connecteur, vous devez en créer un.
   - Si vous prévoyez de modifier un modèle existant, sélectionnez le nom de ce modèle dans la liste située dans le tableau **Template Name** sous **Select the Existing Template to Modify: Find Template**.
   - Ce tableau affiche les noms de tous les modèles disponibles. Vous pouvez également rechercher un modèle.

Indication des caractéristiques générales : Lorsque vous cliquez sur **Next** pour sélectionner un modèle, la boîte de dialogue **Properties - Connector-Specific Property Template** s'affiche. Cette boîte de dialogue contient des onglets pour les caractéristiques générales des propriétés définies et pour les restrictions liées aux valeurs. L'écran général contient les zones suivantes :

- **General:**
  Property Type
  Updated Method
  Description
- **Flags**
  Standard flags
- **Custom Flag**
  Flag

Une fois que vous avez sélectionné les caractéristiques générales de la propriété, cliquez sur l'onglet **Value**.

Indication de valeurs : L'onglet **Value** vous permet de définir la longueur maximum, le nombre maximum de valeurs multiples, une valeur par défaut ou un intervalle de valeurs pour la propriété. Il autorise également les valeurs modifiables. Pour ce faire, procédez comme suit :

1. Cliquez sur l'onglet **Value**. Le panneau d'affichage des valeurs remplace le panneau d'affichage général.
2. Sélectionnez le nom de la propriété dans l'écran **Edit properties**.
3. Dans les zones relatives à la **longueur maximum** et au **nombre maximum de valeurs multiples**, entrez les valeurs de votre choix.

Pour créer une valeur de propriété, procédez comme suit :

1. Sélectionnez la propriété dans la liste **Edit properties** et cliquez dessus avec le bouton droit.
2. Dans la boîte de dialogue, cliquez sur **Add**.

3. Entrez le nom de la nouvelle valeur de propriété et cliquez sur OK. La valeur apparaît dans le panneau **Value** situé dans la partie droite.

Le panneau **Value** contient un tableau comprenant trois colonnes :

La colonne **Value** contient la valeur que vous avez entrée dans la boîte de dialogue **Property Value** et toutes les valeurs que vous avez précédemment créées.

La colonne **Default Value** vous permet d'indiquer n'importe quelle valeur comme valeur par défaut.

La colonne **Value Range** contient l'intervalle que vous avez entré dans la boîte de dialogue **Property Value**.

Une fois que vous avez créé une valeur et qu'elle apparaît dans la grille, vous pouvez la modifier dans le tableau.

Pour modifier une valeur existante dans le tableau, sélectionnez une ligne entière en cliquant sur le numéro de ligne. Ensuite, cliquez avec le bouton droit dans la zone **Value** et cliquez sur **Edit Value**.

Définition des dépendances : Une fois les modifications apportées aux onglets **General** et **Value**, cliquez sur **Next**. La boîte de dialogue **Dependencies - Connector-Specific Property Template** s'affiche.

Une propriété dépendante est une propriété qui est incluse dans le modèle et utilisée dans le fichier de configuration *uniquement si* la valeur d'une autre propriété respecte une condition spécifique. Par exemple, `PollQuantity` apparaît dans le modèle uniquement si JMS est le mécanisme de transfert et que `DuplicateEventElimination` a la valeur `True`.
Pour faire en sorte qu'une propriété soit dépendante et définir la condition dont elle dépend, procédez comme suit :

1. Dans l'écran **Available Properties**, sélectionnez la propriété qui doit devenir dépendante.
2. Dans la zone **Select Property**, utilisez le menu déroulant pour sélectionner la propriété qui conservera la valeur conditionnelle.
3. Dans la zone **Condition Operator**, sélectionnez l'une des valeurs suivantes :

   == (égal à)

   != (différent de)

   > (supérieur à)

   < (inférieur à)

   >= (supérieur ou égal à)

   <= (inférieur ou égal à)
4. Dans la zone **Conditional Value**, entrez la valeur requise pour que la propriété dépendante soit incluse dans le modèle.
5. La propriété dépendante est mise en évidence dans l'écran **Available Properties** ; cliquez sur une flèche pour la déplacer vers l'écran **Dependent Property**.
6. Cliquez sur **Finish**. Connector Configurator stocke les informations que vous avez entrées sous la forme d'un document XML, sous `\data\app` dans le répertoire `\bin` où vous avez installé Connector Configurator.

Création d'un nouveau fichier de configuration

Lorsque vous créez un fichier de configuration, vous devez lui attribuer un nom et sélectionner un courtier d'intégration.

- Dans la fenêtre System Manager, cliquez avec le bouton droit sur le dossier **Connectors** et sélectionnez **Create New Connector**. Connector Configurator affiche la boîte de dialogue **New Connector**.
- En mode autonome : dans Connector Configurator, sélectionnez **File>New>Connector Configuration**. Dans la fenêtre New Connector, entrez le nom du nouveau connecteur.

Vous devez également sélectionner un courtier d'intégration. Le courtier que vous sélectionnez détermine les propriétés qui apparaîtront dans le fichier de configuration. Pour sélectionner un courtier, procédez comme suit :

- Dans la zone **Integration Broker**, sélectionnez ICS, les courtiers de messages WebSphere ou la connectivité WAS.
- Faites défiler les zones restantes dans la fenêtre **New Connector**, comme indiqué plus loin dans ce chapitre.

Création d'un fichier de configuration à partir d'un modèle spécifique au connecteur

Une fois que vous avez créé un modèle spécifique au connecteur, vous pouvez l'utiliser pour créer un fichier de configuration :

1. Cliquez sur **File>New>Connector Configuration**.
2. La boîte de dialogue **New Connector** contient les zones suivantes :
   - **Name**

     Entrez le nom du connecteur. Les noms font la différence entre les majuscules et les minuscules. Le nom que vous entrez doit être unique et cohérent avec le nom de fichier d'un connecteur installé sur le système.

     Important : Connector Configurator ne contrôle pas l'orthographe du nom que vous entrez. Vous devez vérifier que le nom est correct.
   - **System Connectivity**

     Cliquez sur ICS, Courtiers de messages WebSphere ou WAS.
   - **Select Connector-Specific Property Template**

     Tapez le nom du modèle conçu pour votre connecteur. Les modèles disponibles s'affichent dans l'écran **Template Name**. Lorsque vous sélectionnez un nom dans l'écran Template Name, l'écran **Property Template Preview** affiche les propriétés spécifiques au connecteur qui ont été définies dans ce modèle.

     Sélectionnez le modèle à utiliser et cliquez sur **OK**.
3. Un écran de configuration apparaît pour le connecteur que vous configurez. La barre de titre contient le nom du courtier d'intégration et du connecteur. Vous pouvez entrer les valeurs de toutes les zones pour terminer la définition maintenant, ou enregistrer le fichier et renseigner les zones ultérieurement.
4. Pour enregistrer le fichier, cliquez sur **File>Save>To File** ou sur **File>Save>To Project**. Pour exécuter un enregistrement dans un projet, System Manager doit être en cours d'exécution.
   Si vous enregistrez un fichier, la boîte de dialogue **Save File Connector** apparaît. Sélectionnez `*.cfg` comme type de fichier, vérifiez dans la zone File Name que le nom est correctement orthographié et que sa casse est correcte, accédez au répertoire dans lequel vous souhaitez enregistrer le fichier et cliquez

sur **Save**. L'écran d'état affiché dans le panneau de message de Connector Configurator indique que le fichier de configuration a été créé.

Important : Le nom et le chemin du répertoire que vous avez définis ici doivent correspondre au nom et au chemin du fichier de configuration du connecteur que vous indiquez dans le fichier de démarrage du connecteur.

5. Pour remplir la définition du connecteur, entrez des valeurs dans les zones de chacun des onglets de la fenêtre Connector Configurator, comme décrit plus loin dans ce chapitre.

Utilisation d'un fichier existant

Vous disposez peut-être d'un fichier existant dans un ou plusieurs des formats suivants :

- Fichier de définition du connecteur.
  Il s'agit d'un fichier texte qui répertorie les propriétés et les valeurs par défaut applicables d'un connecteur spécifique. Certains connecteurs possèdent ce fichier dans un répertoire `\repository` fourni dans leur package d'origine (en général, le fichier a l'extension `.txt` ; par exemple, CN_XML.txt pour le connecteur XML).
- Fichier référentiel ICS.
  Les définitions utilisées dans une implémentation ICS précédente du connecteur peuvent être accessibles dans un fichier référentiel qui a été utilisé pour la configuration de ce connecteur. En général, ce type de fichier a l'extension `.in` ou `.out`.
- Fichier de configuration précédent pour le connecteur.
  En général, ce type de fichier a l'extension *.cfg.

Bien que certaines de ces sources de fichier puissent contenir tout ou partie des propriétés spécifiques au connecteur, le fichier de configuration du connecteur ne sera pas complet tant que vous n'aurez pas ouvert le fichier et défini les propriétés, comme décrit plus loin dans ce chapitre.

Pour utiliser un fichier existant afin de configurer un connecteur, vous devez ouvrir le fichier dans Connector Configurator, réviser la configuration et enregistrer de nouveau le fichier.

Pour ouvrir un fichier *.txt, *.cfg ou *.in dans un répertoire, procédez comme suit :

1. Dans Connector Configurator, cliquez sur **File>Open>From File**.
2. Dans la boîte de dialogue **Open File Connector**, sélectionnez l'un des types de fichier suivants pour afficher les fichiers disponibles :
   - Configuration (`*.cfg`)
   - Référentiel ICS (`*.in`, `*.out`)

     Sélectionnez cette option si vous avez utilisé un fichier référentiel pour configurer le connecteur dans un environnement ICS. Un fichier référentiel peut contenir plusieurs définitions de connecteur, qui apparaissent toutes lorsque vous ouvrez ce fichier.
   - Tous les fichiers (*.*)

     Sélectionnez cette option si un fichier `*.txt` a été fourni dans le package de l'adaptateur pour le connecteur ou qu'un fichier de définition avec une autre extension est disponible.

3. Dans l'écran du répertoire, accédez au fichier de définition du connecteur approprié, sélectionnez-le et cliquez sur **Open**.

Pour ouvrir une configuration de connecteur à partir d'un projet System Manager, procédez comme suit :

1. Démarrez System Manager. Vous pouvez ouvrir une configuration dans System Manager ou l'y enregistrer uniquement si vous avez démarré System Manager.
2. Démarrez Connector Configurator.
3. Cliquez sur **File>Open>From Project**.

Remplissage d'un fichier de configuration

Lorsque vous ouvrez un fichier de configuration ou un connecteur à partir d'un projet, la fenêtre Connector Configurator affiche l'écran de configuration qui contient les valeurs et les attributs courants.

Le titre de l'écran de configuration affiche le courtier d'intégration et le nom du connecteur, comme indiqué dans le fichier. Vérifiez que votre courtier est correct. Dans le cas contraire, modifiez la valeur du courtier avant de configurer le connecteur. Pour ce faire, procédez comme suit :

1. Dans l'onglet **Standard Properties**, sélectionnez la zone de valeur pour la propriété `BrokerType`. Dans le menu déroulant, sélectionnez la valeur `ICS`, `WMQI` ou `WAS`.
2. L'onglet Standard Properties affiche les propriétés associées au courtier sélectionné. Vous pouvez enregistrer le fichier maintenant ou renseigner les autres zones relatives à la configuration, comme décrit dans «Indication des définitions d'objet métier prises en charge», à la page 20.
3. Une fois la configuration terminée, cliquez sur **File>Save>To Project** ou sur **File>Save>To File**.

   Si vous enregistrez dans un fichier, sélectionnez `*.cfg` comme extension, sélectionnez l'emplacement correct pour le fichier et cliquez sur **Save**.

   Si plusieurs configurations de connecteur sont ouvertes, cliquez sur **Save All to File** pour enregistrer toutes les configurations dans un fichier ou cliquez sur **Save All to Project** pour enregistrer toutes les configurations du connecteur dans un projet System Manager.

   Avant d'enregistrer le fichier, Connector Configurator vérifie que vous avez défini des valeurs pour toutes les propriétés standard requises. Si vous n'avez pas défini de valeur pour l'une des propriétés standard requises, Connector Configurator affiche un message indiquant l'échec de la validation. Vous devez attribuer une valeur à la propriété pour pouvoir enregistrer le fichier de configuration.

Définition des propriétés du fichier de configuration

Lorsque vous créez et que vous nommez un nouveau fichier de configuration du connecteur, ou que vous ouvrez un fichier de configuration existant du connecteur, Connector Configurator affiche un écran de configuration avec des onglets pour les catégories des valeurs de configuration requises.

Connector Configurator requiert des valeurs pour les propriétés dans ces catégories pour les connecteurs s'exécutant sur tous les courtiers :

- Propriétés standard
- Propriétés spécifiques au connecteur
- Objets métier pris en charge

- Valeurs des fichiers journaux/fichiers de trace
- Gestionnaire de données (applicable pour les connecteurs qui utilisent la messagerie JMS avec une livraison des événements garantie)

Remarque : Pour les connecteurs utilisant la messagerie JMS, une catégorie supplémentaire peut s'afficher ; elle est associée à la configuration des gestionnaires de données qui convertissent les données en objets métier.

Pour les connecteurs qui s'exécutent sur **ICS**, des valeurs sont également requises pour ces propriétés :

- Mappes associées
- Ressources
- Messagerie (le cas échéant)

Important : Connector Configurator accepte des valeurs de propriété dans des jeux de caractères anglais et des jeux de caractères qui n'existent pas en anglais. Cependant, les noms des propriétés standard et des propriétés spécifiques au connecteur ainsi que les noms des objets métier pris en charge doivent uniquement utiliser le jeu de caractères anglais.

Les différences entre les propriétés standard et les propriétés spécifiques au connecteur sont les suivants :

- Les propriétés standard d'un connecteur sont partagées par le composant spécifique à l'application d'un connecteur et son courtier. Tous les connecteurs ont le même jeu de propriétés standard. Ces propriétés sont décrites dans l'Annexe A de chaque guide de l'adaptateur. Vous pouvez modifier une partie de ces valeurs uniquement.
- Les propriétés spécifiques à l'application s'appliquent uniquement au composant spécifique à l'application d'un connecteur, c'est-à-dire au composant qui interagit directement avec l'application. Chaque connecteur a des propriétés spécifiques à l'application qui sont propres à cette application. Certaines de ces propriétés fournissent des valeurs par défaut, et d'autres non ; vous pouvez modifier certaines des valeurs par défaut. Les chapitres relatifs à l'installation et à la configuration de chaque guide de l'adaptateur décrivent les propriétés spécifiques à l'application et les valeurs recommandées.

Les zones relatives aux **propriétés standard** et aux **propriétés spécifiques au connecteur** sont codées en couleur pour indiquer les éléments configurables :

- Une zone avec un arrière-plan gris indique une propriété standard. Vous pouvez modifier la valeur, mais vous ne pouvez pas modifier le nom ou supprimer la propriété.
- Une zone avec un arrière-plan blanc indique une propriété spécifique à l'application. Ces propriétés varient en fonction des besoins spécifiques de l'application ou du connecteur. Vous pouvez modifier la valeur et supprimer ces propriétés.
- Les zones de valeurs sont configurables.
- La zone **Update Method** s'affiche pour chaque propriété. Elle indique si le redémarrage d'un composant ou d'un agent est nécessaire pour activer les valeurs modifiées. Vous ne pouvez pas configurer ce paramètre.

Configuration des propriétés standard du connecteur

Les propriétés de configuration standard fournissent des informations utilisées par tous les connecteurs. Voir «Propriétés de configuration standard pour les connecteurs», à la page 53 pour plus d'informations sur ces propriétés.

Important : Ce connecteur prenant en charge tous les courtiers d'intégration, les propriétés de configuration de tous les courtiers lui sont applicables.

Vous devez définir au moins les propriétés standard de configuration du connecteur avant de l'exécuter :

- AgentTraceLevel
- ApplicationName
- ControllerStoreAndForwardMode
- ControllerTraceLevel
- DeliveryTransport

Pour modifier la valeur d'une propriété standard, procédez comme suit :

1. Cliquez dans la zone dont vous souhaitez définir la valeur.
2. Entrez une valeur ou sélectionnez-en une dans le menu déroulant le cas échéant.
3. Une fois que vous avez entré toutes les valeurs pour les propriétés standard, vous pouvez exécuter les opérations suivantes :
   - Pour ignorer les modifications, conserver les valeurs d'origine et quitter Connector Configurator, cliquez sur **File>Exit** (ou fermez la fenêtre) et cliquez sur **No** lorsqu'un message vous demande si vous souhaitez enregistrer les modifications.
   - Pour entrer les valeurs des autres catégories dans Connector Configurator, sélectionnez l'onglet relatif à la catégorie. Les valeurs que vous entrez pour la catégories **Standard Properties** (ou n'importe quelle autre catégorie) sont conservées lorsque vous passez à la catégorie suivante. Lorsque vous fermez la fenêtre, vous êtes invité à enregistrer ou à annuler les valeurs que vous avez entrées dans toutes les catégories.
   - Pour enregistrer les valeurs révisées, cliquez sur **File>Exit** (ou fermez la fenêtre) et sur **Yes** lorsqu'un message vous demande si vous souhaitez enregistrer les modifications. Vous pouvez également cliquer sur **Save>To File** dans le menu File ou la barre d'outils.

Propriétés spécifiques au connecteur : Les propriétés de configuration spécifiques au connecteur fournissent des informations requises par le connecteur au moment de l'exécution. Les propriétés spécifiques au connecteur permettent également de modifier les informations statiques ou logiques dans l'agent du connecteur sans devoir le recoder et le recompiler.

Le Tableau 2 dresse la liste des propriétés de configuration spécifiques au connecteur. Pour obtenir des explications sur les propriétés, voir la section suivante.

Tableau 2. les propriétés spécifiques au connecteur.

Nom	Valeurs possibles	Valeur par défaut	Obligatoire ?
ApplicationName	`iSeriesAdapter`	Aucun	Oui
UseDefaults	*valeur par défaut*	Aucun	Oui
MessageFileName	`BIA_iSeriesAdapter.txt`	`BIA_iSeriesAdapter.txt`	Non
PollQuantity	un entier supérieur à 1	1	Non

ApplicationName : Nom unique devant être précisé pour chaque connecteur.

UseDefaults : Par exemple, certains paramètres d'entrée d'un programme peuvent être des constantes. Ces attributs peuvent donc être conçus pour avoir des valeurs par défaut. En l'absence de valeur par défaut, et si la propriété UseDefaults est définie sur `true`, l'adaptateur génère une erreur et émet un message d'erreur VerbProcessingFailedException. Si UseDefaults n'est pas défini ou s'il est défini sur `false`, et en l'absence de valeurs par défaut, l'adaptateur génère une chaîne de longueur MaxLength complétée d'espaces vides pour les valeurs de l'attribut.

MessageFileName : Nom et chemin du fichier de message d'erreur, s'il n'est pas situé à l'emplacement standard de message %CROSSWORLDS%\ connectors\messages. Si le nom de fichier du message n'est pas un chemin qualifié complet, il est supposé que le fichier figure dans le répertoire indiqué par la variable d'environnement HOME ou par le paramètre de démarrage user.home. S'il n'existe pas de fichier de message de connecteur, le fichier BIA_iSeriesAdapter.txt est utilisé comme fichier de message.

PollQuantity : PollQuantity est un entier supérieur à 1, qui indique le nombre d'éléments à interroger à partir des files d'attente de données. Notez que si `n` est précisé par PollQuantity, alors chaque file d'attente configurée à l'aide des méta-objets est interrogée `n` fois. La valeur par défaut est 1.

Définition de propriétés de configuration spécifiques à l'application

Pour les propriétés de configuration spécifiques à l'application, vous pouvez ajouter ou modifier les noms des propriétés, définir des valeurs, supprimer une propriété et chiffrer une propriété. La longueur par défaut d'une propriété est de 255 caractères.

1. Cliquez avec le bouton droit dans la partie supérieure gauche de la grille. Une barre de menus contextuelle apparaît. Cliquez sur **Add** pour ajouter une propriété. Pour ajouter une propriété enfant, cliquez avec le bouton droit sur le numéro de la ligne parent et cliquez sur **Add child**.
2. Entrez une valeur pour la propriété ou la propriété enfant.
3. Pour chiffrer une propriété, cochez la case **Encrypt**.
4. Vous pouvez enregistrer ou ignorer les modifications, comme décrit pour «Configuration des propriétés standard du connecteur», à la page 18.

La zone Update Method affichée pour chaque propriété indique si le redémarrage d'un composant ou d'un agent est nécessaire à l'activation des valeurs modifiées.

Important : La modification du nom prédéfini d'une propriété de connecteur spécifique à l'application peut entraîner l'échec d'un connecteur. Le connecteur peut nécessiter certains noms de propriété pour se connecter à une application ou s'exécuter correctement.

Chiffrement des propriétés du connecteur : Pour chiffrer les propriétés spécifiques à l'application, cochez la case **Encrypt** dans la fenêtre des propriétés spécifiques au connecteur. Pour déchiffrer une valeur, décochez la case **Encrypt**, entrez la valeur appropriée dans la boîte de dialogue **Verification** et cliquez sur **OK**. Si la valeur entrée est correcte, elle est déchiffrée et s'affiche.

Le guide d'utilisateur de l'adaptateur pour chaque connecteur contient la liste et la description de chaque propriété ainsi que sa valeur par défaut.

Si une propriété a plusieurs valeurs, la case **Encrypt** apparaît pour la première valeur de la propriété. Lorsque vous sélectionnez **Encrypt**, toutes les valeurs de la propriété sont chiffrées. Pour déchiffrer plusieurs valeurs d'une propriété, décochez la case **Encrypt** pour la première valeur de la propriété, puis entrez la nouvelle valeur dans la boîte de dialogue **Verification**. Si la valeur entrée est une correspondance, toutes les valeurs multiples sont déchiffrées.

Méthode de mise à jour : Reportez-vous aux descriptions des méthodes de mise à jour contenues dans l'annexe *Propriétés de configuration standard pour les connecteurs* dans «Présentation des valeurs des propriétés de configuration», à la page 54.

Indication des définitions d'objet métier prises en charge

Utilisez l'onglet **Supported Business Objects** dans Connector Configurator pour indiquer les objets métier que le connecteur utilisera. Vous devez indiquer les objets métier génériques et les objets métier spécifiques à l'application, et indiquer les associations pour les mappes entre les objets métier.

Remarque : Certains connecteurs nécessitent que des objets métier soient indiqués comme étant pris en charge pour pouvoir exécuter la notification des événements ou une configuration supplémentaire (à l'aide des méta-objets) avec leurs applications. Pour plus d'informations, voir *Connector Development Guide for C++* ou *Connector Development Guide for Java*.

Si ICS est votre courtier : Pour indiquer qu'une définition d'objet métier est prise en charge par le connecteur ou modifier les paramètres de prise en charge d'une définition d'objet métier existante, cliquez sur l'onglet **Supported Business Objects** et utilisez les zones suivants :

Nom de l'objet métier : Pour indiquer qu'une définition d'objet métier est prise en charge par le connecteur, avec System Manager en cours d'exécution, procédez comme suit :

1. Cliquez dans une zone vide de la liste **Business Object Name**. Une boîte à liste déroulante affiche toutes les définitions d'objet métier qui existent dans le projet System Manager.
2. Cliquez sur un objet métier pour l'ajouter.
3. Définissez la zone **Agent Support** (décrite plus bas) pour l'objet métier.
4. Dans le menu File de la fenêtre Connector Configurator, cliquez sur **Save to Project**. La définition révisée du connecteur, qui contient la prise en charge sélectionnée pour la définition de l'objet métier ajouté, est enregistrée dans un projet ICL (Integration Component Library) de System Manager.

Pour supprimer un objet métier dans la liste des objets métier pris en charge :

1. Pour sélectionner la zone d'un objet métier, cliquez sur le numéro situé à gauche de l'objet métier.

2. Dans le menu **Edit** de la fenêtre Connector Configurator, cliquez sur **Delete Row**. L'objet métier est supprimé de la liste.
3. Dans le menu **File**, cliquez sur **Save to Project**.

La suppression d'un objet métier dans la liste des objets métier pris en charge modifie la définition du connecteur et rend l'objet métier supprimé inutilisable dans cette implémentation du connecteur. Elle n'affecte pas le code du connecteur et ne supprime pas la définition de l'objet métier dans System Manager.

Prise en charge de l'agent : Si un objet métier dispose de la prise en charge de l'agent, le système tente d'utiliser cet objet métier pour fournir des données à une application via l'agent du connecteur.

En général, les objets métier spécifiques à l'application pour un connecteur sont pris en charge par l'agent de ce connecteur, mais les objets métier génériques ne le sont pas.

Pour indiquer que l'objet métier est pris en charge par l'agent du connecteur, cochez la case **Agent Support**. La fenêtre Connector Configurator ne valide pas vos sélections pour Agent Support.

Niveau de transaction maximum : Le niveau de transaction maximum d'un connecteur correspond au niveau de transaction le plus élevé pris en charge par le connecteur.

Pour la plupart des connecteurs, `Best Effort` est la seule valeur possible.

Vous devez redémarrer le serveur pour que les modifications prennent effet.

Si votre courtier est un courtier de messages WebSphere : Si vous utilisez le mode autonome (non connecté à System Manager), vous devez entrer manuellement le nom de l'objet métier.

Si System Manager est en cours d'exécution, vous pouvez cocher la case située sous la colonne **Business Object Name** dans l'onglet **Supported Business Objects**. Une boîte de dialogue mixte affiche la liste des objets métier disponibles dans le projet Integration Component Library auquel le connecteur appartient. Dans cette liste, sélectionnez l'objet métier de votre choix.

La zone **Message Set ID** est facultative pour WebSphere Business Integration Message Broker 5.0, et sa valeur ne doit pas nécessairement être unique le cas échéant. Cependant, pour WebSphere MQ Integrator et Integrator Broker 2.1, vous devez indiquer un **ID** unique.

Si WAS est votre courtier : Lorsque vous sélectionnez WebSphere Application Server comme type de courtier, Connector Configurator ne nécessite pas les ID d'ensemble de messages. L'onglet **Supported Business Objects** contient la colonne **Business Object Name** pour les objets métier pris en charge uniquement.

Si vous utilisez le mode autonome (non connecté à System Manager), vous devez entrer manuellement le nom de l'objet métier.

Si System Manager est en cours d'exécution, vous pouvez cocher la case située sous la colonne Business Object Name dans l'onglet Supported Business Objects. Une boîte de dialogue mixte affiche la liste des objets métier disponibles dans le

projet Integration Component Library auquel le connecteur appartient. Dans cette liste, sélectionnez l'objet métier de votre choix.

Mappes associées (ICS uniquement)

Chaque connecteur prend en charge la liste des définitions des objets métier et leurs mappes associées actives dans WebSphere InterChange Server. Cette liste apparaît lorsque vous sélectionnez l'onglet **Associated Maps**.

La liste des objets métier contient l'objet métier spécifique à l'application pris en charge par l'agent et l'objet générique correspondant que le contrôleur envoie à la collaboration de souscription. L'association d'une mappe détermine la mappe qui sera utilisée pour transformer l'objet métier spécifique à l'application en objet métier générique, ou inversement.

Si vous utilisez des mappes uniquement définies pour des objets métier source et cible spécifiques, les mappes sont déjà associées aux objets métier appropriés lorsque vous affichez l'écran, et vous n'avez pas besoin de (ou ne pouvez pas) les modifier.

Si plusieurs mappes sont disponibles pour un objet métier pris en charge, vous devez lier de manière explicite cet objet métier à la mappe qu'il doit utiliser.

L'onglet **Associated Maps** affiche les zones suivantes :

- **Business Object Name**

  Il s'agit des objets métier pris en charge par ce connecteur, comme indiqué dans l'onglet **Supported Business Objects**. Si vous indiquez des objets métier supplémentaires dans l'onglet Supported Business Objects, ils sont reflétés dans cette liste une fois que vous avez enregistré les modifications en sélectionnant **Save to Project** dans le menu **File** de la fenêtre Connector Configurator.
- **Associated Maps**

  L'écran affiche toutes les mappes installées sur le système à utiliser avec les objets métier pris en charge du connecteur. L'objet métier source pour chaque mappe s'affiche à gauche du nom de la mappe, dans l'écran **Business Object Name**.
- **Explicit**

  Dans certains cas, vous devez peut-être lier de manière explicite une mappe associée.

  Une liaison explicite est requise uniquement lorsque plusieurs mappes existent pour un objet métier pris en charge spécifique. Lorsque ICS s'amorce, il tente de lier automatiquement une mappe à chaque objet métier pris en charge pour chacun des connecteurs. Si plusieurs mappes prennent le même objet métier comme entrée, le serveur tente de localiser et de lier une mappe qui correspond au sur-ensemble des autres.

  Si aucune mappe n'est le sur-ensemble des autres, le serveur ne peut pas lier l'objet métier à une seule mappe et vous devrez définir la liaison de manière explicite.

  Pour lier une mappe de manière explicite, procédez comme suit :

  1. Dans la colonne **Explicit**, cochez la case correspondant à la mappe à lier.
  2. Sélectionnez la mappe que vous souhaitez associer à l'objet métier.
  3. Dans le menu **File** de la fenêtre Connector Configurator, cliquez sur **Save to Project**.
  4. Déployez le projet jusqu'à ICS.
  5. Réamorcez le serveur pour que les modifications prennent effet.

Ressources (ICS)

L'onglet **Resource** vous permet de définir une valeur qui détermine si l'agent du connecteur gérera plusieurs processus simultanément, et dans quelle mesure, à l'aide du parallélisme de l'agent du connecteur.

Tous les connecteurs ne prennent pas en charge cette fonction. Si vous exécutez un agent de connecteur conçu dans Java pour être multithread, nous vous recommandons de ne pas utiliser cette fonction dans la mesure où il est généralement plus efficace d'utiliser plusieurs unités d'exécution plutôt que plusieurs processus.

Messagerie (ICS)

Les propriétés de messagerie sont disponibles uniquement si vous avez défini MQ comme la valeur de la propriété standard `DeliveryTransport` et ICS comme le type de courtier. Ces propriétés affectent la manière dont le connecteur utilisera les files d'attente.

Définition des valeurs du fichier de trace ou du fichier journal

Lorsque vous ouvrez le fichier de configuration ou le fichier de définitions d'un connecteur, Connector Configurator utilise les valeurs de journalisation et de trace de ce fichier comme valeurs par défaut. Vous pouvez modifier ces valeurs dans Connector Configurator.

Pour modifier les valeurs de journalisation et de trace, procédez comme suit :

1. Cliquez sur l'onglet **Trace/Log Files**.
2. Pour la journalisation ou la fonction de trace, vous pouvez écrire des messages à l'un des composants suivants :
   - A la console (STDOUT) :
     Ecrit des messages de journalisation ou de trace à l'écran STDOUT.

     Remarque : Vous pouvez utiliser l'option STDOUT de l'onglet **Trace/Log Files** pour les connecteurs s'exécutant sur la plate-forme Windows.
   - A un fichier :
     Ecrit des messages de journalisation ou de trace vers un fichier indiqué. Pour indiquer le fichier, cliquez sur le bouton du répertoire (ellipse), accédez à l'emplacement de votre choix, indiquez un nom de fichier et cliquez sur **Save**. Les messages de journalisation ou de trace sont écrits vers le fichier et l'emplacement indiqués.

     Remarque : Les fichiers de journalisation et de trace sont de simples fichiers texte. Vous pouvez utiliser l'extension de votre choix lorsque vous définissez les noms de fichier. Cependant, pour les fichiers de trace, nous vous recommandons d'utiliser l'extension `.trace` plutôt que l'extension `.trc`, afin d'éviter toute confusion avec les autres fichiers pouvant résider sur le système. Pour les fichiers de journalisation, les extensions classiques sont `.log` et `.txt`.

Gestionnaires de données

La section des gestionnaires de données est disponible pour la configuration uniquement si vous avez indiqué une valeur JMS pour ContainerManagedEvents. Tous les adaptateurs n'utilisent pas les gestionnaires de données.

Pour connaître les valeurs à utiliser pour ces propriétés, reportez-vous aux descriptions sous `ContainerManagedEvents` dans l'Annexe A, Propriétés standard. Pour plus d'informations, voir *Connector Development Guide for C++* ou *Connector Development Guide for Java*.

Enregistrement du fichier de configuration

Une fois que vous avez configuré votre connecteur, enregistrez son fichier de configuration. Connector Configurator enregistre le fichier dans le mode courtier que vous avez sélectionné pendant la configuration. La barre de titre de Connector Configurator affiche toujours le mode courtier (ICS, WMQI ou WAS) en cours d'utilisation.

Le fichier est enregistré en tant que document XML. Pour enregistrer le document XML, vous avez trois possibilités :

- dans System Manager, en tant que fichier avec l'extension `*.con` dans le projet ICL ;
- dans un répertoire que vous avez indiqué ;
- en mode autonome, en tant que fichier avec l'extension `*.cfg` dans un répertoire (par défaut, le fichier est enregistré dans `\WebSphereAdapters\bin\Data\App`) ;
- dans un projet WebSphere Application Server, le cas échéant.

Pour plus d'informations sur l'utilisation des projets dans System Manager et sur le déploiement, voir les guides d'implémentation suivants :

- Pour ICS : *Implementation Guide for WebSphere InterChange Server*
- Pour les courtiers de messages WebSphere : *Implementing Adapters with WebSphere Message Brokers*
- Pour WAS : *Implementing Adapters with WebSphere Application Server*

Modification d'un fichier de configuration

Vous pouvez modifier les paramètres du courtier d'intégration pour un fichier de configuration existant. Cela vous permet d'utiliser le fichier comme modèle pour la création d'un nouveau fichier de configuration que vous pouvez utiliser avec un autre courtier.

Remarque : Vous devrez modifier d'autres propriétés de configuration ainsi que la propriété du mode courtier si vous changez de courtiers d'intégration.

Pour modifier votre sélection de courtier dans un fichier de configuration existant (facultatif) :

- Ouvrez le fichier de configuration existant dans Connector Configurator.
- Sélectionnez l'onglet **Standard Properties**.
- Dans la zone **BrokerType** de l'onglet Standard Properties, sélectionnez la valeur appropriée pour votre courtier.
  Lorsque vous modifiez la valeur courante, les onglets et les zones disponibles dans l'écran des propriétés changent immédiatement, et seules les onglets et les zones appartenant au nouveau courtier sélectionné apparaissent.

Exécution de la configuration

Une fois que vous avez créé un fichier de configuration pour un connecteur et que vous l'avez modifié, assurez-vous que le connecteur peut localiser le fichier de configuration lorsqu'il démarre.

Pour ce faire, ouvrez le fichier de démarrage utilisé pour le connecteur et vérifiez que le nom de fichier et l'emplacement utilisés pour le fichier de configuration du connecteur correspondent exactement au nom attribué au fichier et au répertoire ou au chemin d'accès dans lequel vous l'avez placé.

Utilisation de Connector Configurator en environnement globalisé

Connector Configurator est globalisé et peut gérer la conversion des caractères entre le fichier de configuration et le courtier d'intégration. Connector Configurator utilise le codage natif. Lorsqu'il écrit dans le fichier de configuration, il utilise le codage UTF-8.

Connector Configurator prend en charge les caractères qui n'existent pas en anglais dans :

- toutes les zones de valeur ;
- le chemin d'accès au fichier journal et au fichier de trace (indiqué dans l'onglet **Trace/Log files**).

La liste déroulante pour les propriétés de configuration standard `CharacterEncoding` et `Locale` affiche uniquement un sous-ensemble des valeurs prises en charge. Pour ajouter d'autres valeurs à cette liste, vous devez modifier manuellement le fichier `\Data\Std\stdConnProps.xml` dans le répertoire produit.

Par exemple, pour ajouter l'environnement local `en_GB` à la liste des valeurs pour la propriété `Locale`, ouvrez le fichier `stdConnProps.xml` et ajoutez la ligne en caractère gras comme indiqué ci-dessous :

```
<Property name="Locale"
isRequired="true"
updateMethod="component restart">
                <ValidType>String</ValidType>
            <ValidValues>
                                <Value>ja_JP</Value>
                                <Value>ko_KR</Value>
                                <Value>zh_CN</Value>
                                <Value>zh_TW</Value>
                                <Value>fr_FR</Value>
                                <Value>de_DE</Value>
                                <Value>it_IT</Value>
                                <Value>es_ES</Value>
                                <Value>pt_BR</Value>
                                <Value>en_US</Value>
                                <Value>en_GB</Value>
                        <DefaultValue>en_US</DefaultValue>
            </ValidValues>
    </Property>
```

Démarrage du connecteur

Vous devez démarrer un connecteur de manière explicite à l'aide du **script de démarrage du connecteur**. Sous Windows, le script de démarrage doit résider dans le répertoire d'exécution du connecteur :

ProductDir\connectors*connName*

où *connName* identifie le connecteur.

Sous UNIX, le script de démarrage doit résider dans le répertoire *UNIX ProductDir*/bin.

Le nom du script de démarrage dépend de la plateforme du système d'exploitation, comme le montre le tableau 3.

Tableau 3. Script de démarrage pour un connecteur

Système d'exploitation	Script de démarrage
Systèmes UNIX	`connector_manager`
Windows	`start_connName.bat`

Lorsque le script de démarrage s'exécute, il va chercher par défaut le fichier de configuration dans le *Productdir* (voir commandes ci-dessous). Il s'agit du répertoire dans lequel vous placez le fichier de configuration.

Remarque : Si l'adaptateur utilise le transfert JMS, vous avez besoin d'un fichier de configuration local .

Pour appeler le script de démarrage du connecteur, utilisez l'une des méthodes suivantes :

- Sur les systèmes Windows, dans le menu **Démarrer** :

  Sélectionnez **Programmes>IBM WebSphere Business Integration Adapters>Adapters>Connectors**. Par défaut, le nom du programme est "IBM WebSphere Business Integration Adapters". Cependant, vous pouvez le personnaliser. Vous pouvez également créer sur le bureau un raccourci vers le connecteur.
- A partir de la ligne de commande :
  - Sur les systèmes Windows :

    `start_connName connName brokerName [-cconfigFile]`
  - Sur les systèmes UNIX :

    `connector_manager -start connName brokerName [-cconfigFile ]`

  où *connName* est le nom du connecteur et *brokerName* le nom de votre connecteur d'intégration, comme suit :
  - Pour WebSphere InterChange Server, indiquez à la place de *brokerName* le nom de l'instance ICS.
  - Pour les courtiers de messages WebSphere (WebSphere MQ Integrator, WebSphere MQ Integrator Broker ou WebSphere Business Integration Message Broker) ou WebSphere Application Server, remplacez *brokerName* par une chaîne identifiant le courtier.

    Remarque : Pour un courtier de messages WebSphere ou WebSphere Application Server résidant sur un système Windows, vous *devez* inclure l'option `-c` suivie du nom du fichier de configuration du connecteur. Pour ICS, l'option `-c` est facultative.
- A partir de Adapter Monitor, qui est lancé au démarrage de System Manager, lequel est exécuté avec le courtier WebSphere Application Server ou InterChange Server :

  Vous pouvez charger, activer, désactiver, interrompre, arrêter ou supprimer un connecteur à l'aide de cet outil.
- A partir de System Manager (pour tous les courtiers) :

  Vous pouvez charger, activer, désactiver, interrompre, arrêter ou supprimer un connecteur à l'aide de cet outil.

- Sur les systèmes Windows, vous pouvez configurer le connecteur de sorte qu'il démarre comme un service Windows. Dans ce cas, le connecteur démarre lors de l'amorçage du système Windows (pour un service automatique) ou lorsque vous démarrez le service via la fenêtre Services Windows (pour un service manuel).

Pour plus d'informations sur le démarrage d'un connecteur, notamment sur les options de lancement à partir de la ligne de commande, reportez-vous à l'un des documents suivants :

- Pour WebSphere InterChange Server, voir *System Administration Guide*.
- Pour les courtiers de messages WebSphere, voir *Implementing Adapters with WebSphere Message Brokers*.
- Pour WebSphere Application Server, voir *Implementing Adapters with WebSphere Application Server*.

Arrêt du connecteur

La méthode pour arrêter un connecteur dépend de la manière dont il a été démarré, comme suit :

- Si vous avez démarré le connecteur à partir d'une ligne de commande, avec son script de démarrage :
  - Sur les systèmes Windows, l'appel du script de démarrage crée une fenêtre de "console" séparée pour le connecteur. Dans cette fenêtre, tapez "Q" et appuyez sur Entrée pour arrêter le connecteur.
  - Avec InterChange Server sur les systèmes UNIX, les connecteurs s'exécutent en arrière-plan de sorte qu'ils n'ont pas de fenêtre séparée. Vous devez exécuter la commande suivante pour arrêter le connecteur :

    ```
    connector_manager_connName -stop
    ```

    où *connName* correspond au nom du connecteur.
- A partir d'Adapter Monitor (produit WebSphere Business Integration Adapters uniquement), lancé lorsque vous démarrez System Manager :

  Vous pouvez charger, activer, désactiver, interrompre, arrêter ou supprimer un connecteur à l'aide de cet outil.
- A partir de System Monitor (produit WebSphere InterChange Server uniquement) :

  Vous pouvez charger, activer, désactiver, interrompre, arrêter ou supprimer un connecteur à l'aide de cet outil.
- Sur les systèmes Windows, vous pouvez configurer le connecteur de sorte qu'il démarre comme un service Windows. Dans ce cas, le connecteur s'arrête en même temps que le système Windows.

Création de plusieurs instances de connecteur

La création de plusieurs instances d'un connecteur revient pratiquement à créer un connecteur personnalisé. Vous pouvez configurer votre système de sorte qu'il crée et exécute plusieurs instances d'un connecteur en suivant les étapes ci-dessous. Vous devez :

- créer un répertoire pour l'instance du connecteur ;
- vérifier que vous possédez les définitions d'objet métier requises ;
- créer un fichier de définition pour le connecteur ;
- créer un script de démarrage.

Création d'un répertoire

Vous devez créer un répertoire pour chaque instance de connecteur. Vous devez attribuer le nom suivant à ce répertoire de connecteur :

```
ProductDir\connectors\connectorInstance
```

où `connectorInstance` identifie de manière unique l'instance de connecteur.

Si le connecteur possède des méta-objets qui lui sont spécifiques, vous devez créer un méta-objet pour l'instance de connecteur. Si vous enregistrez le méta-objet en tant que fichier, créez le répertoire suivant et stockez le fichier dedans :

```
ProductDir\repository\connectorInstance
```

Création de définitions d'objet métier : Si les définitions d'objet métier pour chaque instance du connecteur n'existent pas déjà dans le projet, vous devez les créer.

1. Si vous devez modifier les définitions d'objet métier associées au connecteur initial, copiez les fichiers appropriés et utilisez Business Object Designer pour les importer. Vous pouvez copier n'importe quel fichier pour le connecteur initial. Vous devez simplement les renommer si vous les modifiez.
2. Les fichiers pour le connecteur initial doivent résider dans le répertoire suivant :

   ```
   ProductDir\repository\initialConnectorInstance
   ```

   Tous les fichiers supplémentaires que vous créez doivent être placés dans le sous-répertoire `connectorInstance` approprié de `ProductDir\repository`.

Création d'une définition de connecteur : Vous devez créer un fichier de configuration (définition du connecteur) pour l'instance du connecteur dans Connector Configurator. Pour ce faire, procédez comme suit :

1. Copiez le fichier de configuration du connecteur initial (définition du connecteur) et renommez-le.
2. Assurez-vous que chaque instance du connecteur répertorie correctement ses objets métier pris en charge (et tous les méta-objets associés).
3. Personnalisez toutes les propriétés du connecteur le cas échéant.

Création d'un script de démarrage : Pour créer un script de démarrage, procédez comme suit :

1. Copiez le script de démarrage du connecteur initial et attribuez-lui un nom incluant le nom du répertoire du connecteur :

   dirname
2. Placez ce script de démarrage dans le répertoire du connecteur créé à la section «Création d'un répertoire».
3. Créez un raccourci pour le script de démarrage (Windows uniquement).
4. Copiez le texte du raccourci du connecteur et modifiez le nom du connecteur initial (dans la ligne de commande) de sorte qu'il corresponde au nom de la nouvelle instance du connecteur.

A présent, vous pouvez exécuter simultanément les deux instances du connecteur sur votre serveur d'intégration.

Pour plus d'informations sur la création de connecteurs personnalisés, voir *Connector Development Guide for C++ or for Java*.

Chapitre 4. Présentation des objets métier pour le connecteur

Le présent chapitre décrit la structure des objets métier iSeries, la façon dont ils sont traités par le connecteur et les suppositions que le connecteur fait à leur sujet . Vous utiliserez ces informations pour modifier les objets métier existants pour iSeries ou pour vous aider à en implémenter de nouveaux.

Ce chapitre contient les sections suivantes :

- «Définition des métadonnées du connecteur»
- «Structure des objets métier pour les programmes RPG, COBOL et Java», à la page 30
- «Structure Business Object pour les files d'attente de données iSeries», à la page 34
- «Configuration des méta-objets en vue de l'interrogation», à la page 34
- «Spécification de propriétés d'attribut d'objet métier», à la page 36
- «Spécification du texte de l'application au niveau de l'attribut de l'objet métier», à la page 37

Pour obtenir des informations sur l'utilitaire Object Discovery Agent (ODA) qui automatise la création d'objets métier pour IBM WebSphere Business Integration Adapter pour iSeries, voir le Chapitre 5, «Création et modification d'objets métier», à la page 39.

Définition des métadonnées du connecteur

Le connecteur iSeries est contrôlé par les métadonnées. Dans le système WebSphere Business Integration, les métadonnées sont des informations spécifiques aux applications. Elles sont conservées dans un objet métier qui facilite les interactions du connecteur avec l'application. Un connecteur contrôlé par métadonnées traite chaque objet métier qu'il prend en charge en fonction des métadonnées codées dans la définition de l'objet métier, plutôt que d'après les instructions codées en dur dans le connecteur. Les métadonnées de l'objet métier incluent sa structure, les valeirs de ses propriétés d'attribut et le contenu de ses informations spécifiques à l'application. Dans la mesure où le connecteur est contrôlé par les métadonnées, il peut traiter de nouveaux objets métier ou des objets modifiés sans avoir à modifier son code.

Le connecteur fait des suppositions sur la structure des objets métier pris en charge et le format des informations spécifiques à l'application.

Par conséquent, lorsque vous créez ou modifiez un objet métier, vos modifications doivent être conformes aux règles que le connecteur est supposé suivre, sinon il ne pourra pas traiter correctement les objets métier nouveaux ou modifiés.

Présentation de la structure de l'objet métier

Dans le système WebSphere Business Integration, une définition d'objet métier se compose d'un nom de type, des verbes pris en charge et des attributs. Un objet métier d'application est une instance d'une définition d'objet métier. Il reflète la structure de données et les propriétés d'attributs d'une application spécifique.

Certains attributs, au lieu de contenir des données, pointent vers les objets métier enfants ou des ensembles d'objets métiers enfants contenant les données.

Les objets métier WebSphere Business Integration Adapter peuvent être plats ou hiérarchiques. Un objet métier plat ne contient que des attributs simples, c'est-à-dire qui représentent une seule valeur (telle qu'une chaîne). Un objet métier hiérarchique contient des attributs simples ainsi que des objets métier enfants ou des tableaux d'objets métier enfants contenant les valeurs.

Pour un objet conteneur, la cardinalité 1 ou la relation de cardinalité simple, indique qu'un attribut contenu dans un objet métier parent ne contient qu'un seul objet métier enfant. Cet objet métier enfant est une collection qui ne peut contenir qu'un seul enregistrement. Le type de l'attribut est le même que celui de l'objet métier enfant.

La cardinalité n, ou la relation de cardinalité multiple, indique qu'un attribut contenu dans un objet métier parent contient un tableau d'objets métier enfants. Dans ce cas, l'objet métier enfant est une collection qui peut contenir plusieurs enregistrements. Le type de l'attribut est le même que celui du tableau d'objets métier enfants.

Un objet métier hiérarchique peut avoir des attributs simples ainsi que des attributs qui représentent un objet métier enfant à cardinalité simple ou un tableau d'objets métier enfants. Chacun de ces objets métier peut à son tour contenir des objets métier enfants à cardinalité simple et des tableaux d'objets métier, et ainsi de suite.

Structure des objets métier pour les programmes RPG, COBOL et Java

L'objet métier de l'adaptateur iSeries est plat. Les attributs peuvent être les paramètres input, output ou inout. Le concepteur de l'objet métier doit utiliser un de ces attributs comme clé.

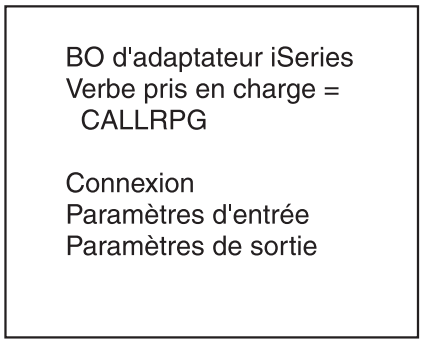


Figure 3. L'objet métier parent iSeries

Il y a également un attribut enfant de type Connection. Il contient des informations sur la connexion avec la machine AS/400, le nom d'hôte, le nom d'utilisateur et le mot de passe. Ces attributs sont tous obligatoires. Par conséquent, `is Required` est vrai pour tous. Cet objet métier Connection est un attribut enfant de tous les objets métier iSeries.

Remarque : Dans l'objet métier iSeries, les informations spécifiques à l'application de l'objet métier de connexion peuvent être vides. Ceci est dû au fait que l'ASI de l'objet métier n'est pas traité par l'adaptateur.

Les informations entre parenthèses dans la figure 5 représentent les informations spécifiques à l'application de l'objet métier.

Remarque : Les noms de verbes ne sont pas sensibles à la casse. Par exemple, le verbe getqueue peut s'écrire GETQUEUE ou GetQueue.

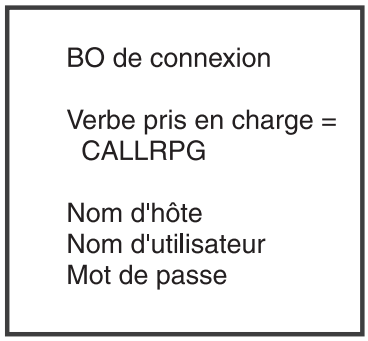


Figure 4. L'objet métier enfant iSeries

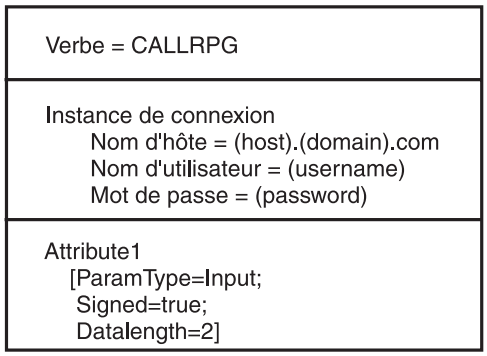
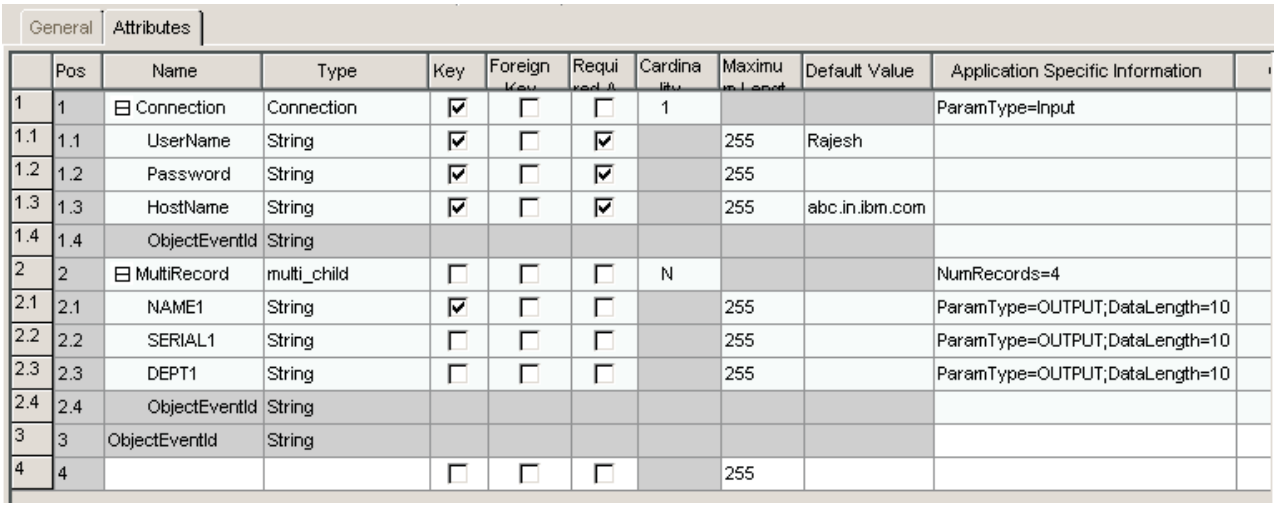


Figure 5. Exemple d'objet métier RPG

L'objet métier RPG est composé du verbe CALLRPG ou CALLPGM, et le nom du programme (chemin complet de IFSFile) est défini en tant qu'ASI du verbe. CALLRPG est utilisé pour appeler les programmes RPG, et CALLPGM pour appeler tout PGM (y compris les programmes RPG, COBOL et Java). Outre l'attribut enfant Connection, l'objet métier RPG possède des attributs correspondants aux paramètres du programme RPG. Le nom de l'attribut est identique à celui du paramètre correspondant indiqué dans le programme source. La propriété MaxLength des attributs (représentant un paramètre RPG) est dérivée de la longueur du paramètre correspondant, précisée dans le PARM Spec du programme source. Si le paramètre est un nombre, la longueur de la partie décimale est également mentionnées dans l'ASI sous la forme `DecimalPositions=n` et `packedDec=true`.

L'adaptateur peut être utilisé pour appeler plusieurs fois un PGM avec un même objet métier Request, mais plusieurs instances. Un exemple est proposé en Figure 6.

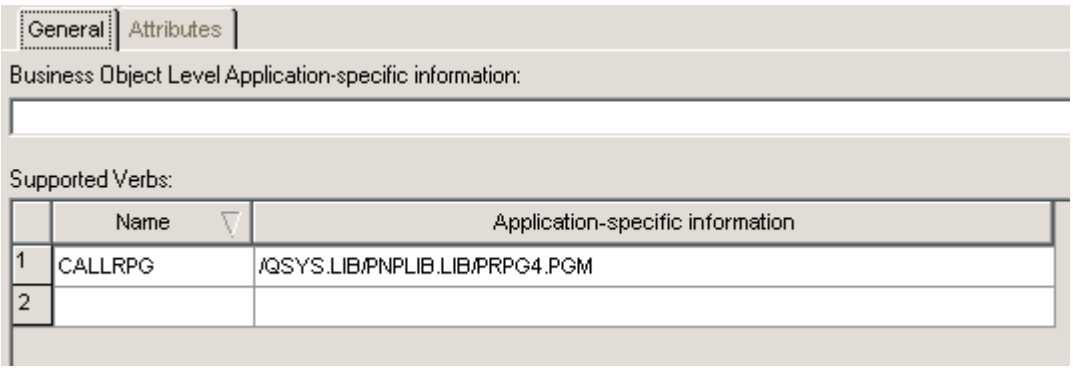


General | Attributes

	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	⊟ Connection	Connection	☑	☐	☐	1			ParamType=Input
1.1	1.1	UserName	String	☑	☐	☑		255	Rajesh	
1.2	1.2	Password	String	☑	☐	☑		255		
1.3	1.3	HostName	String	☑	☐	☑		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	⊟ MultiRecord	multi_child	☐	☐	☐	N			NumRecords=4
2.1	2.1	NAME1	String	☑	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

Figure 6. Un seul objet métier Request, avec plusieurs instances

La Figure 7 représente l'objet métier parent avec ses informations de connexion, et l'objet métier enfant avec les informations sur le paramètre PGM. Les informations sur la connexion figurent dans l'objet métier Connection, et le Verb ASI est identique au chemin du programme à appeler.



General | Attributes

Business Object Level Application-specific information:

Supported Verbs:

	Name	Application-specific information
1	CALLRPG	/QSYS.LIB/PNPLIB.LIB/PRPG4.PGM
2		

Figure 7. Objet métier parent avec informations sur la connexion

L'objet métier enfant est présenté dans la Figure 8 avec des attributs correspondant aux informations de paramètre du programme.

General | Attributes

	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	⊟ Connection	Connection	☑	☐	☐	1			ParamType=Input
1.1	1.1	UserName	String	☑	☐	☑		255	Rajesh	
1.2	1.2	Password	String	☑	☐	☑		255		
1.3	1.3	HostName	String	☑	☐	☑		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	⊟ MultiRecord	multi_child	☐	☐	☐	N			NumRecords=4
2.1	2.1	NAME1	String	☑	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

Figure 8. Objet métier enfant avec informations sur le paramètre correspondant

L'exemple de la Figure 9 présente un objet métier avec deux instances de l'objet métier multi_child. Le programme sera donc exécuté deux fois. Veillez à utiliser les fichiers XSD appropriés pour l'extraction de plusieurs enregistrements.

⊟ Connection	Connection	
UserName	String	Rajesh
Password	String	password
HostName	String	abc.in.ibm.com
ObjectEventId	String	
⊟ MultiRecord[n]	multi_child	
⊟ [0]	multi_child	
NAME1	String	RAJ
SERIAL1	String	10
DEPT1	String	A
ObjectEventId	String	
⊟ [1]	multi_child	
NAME1	String	PRABHU
SERIAL1	String	11
DEPT1	String	B
ObjectEventId	String	
ObjectEventId	String	

Figure 9. Objet métier avec deux instances d'un objet métier multi-child

Traitement de l'objet métier Connector

Le connecteur transmet les objets métier entre le courtier d'intégration et le système AS/400.

Lorsqu'un courtier d'intégration transmet un objet métier au connecteur, celui-ci procède comme suit :

1. Il utilise les informations de l'attribut enfant de la connexion pour se connecter au système AS/400.
2. Il établit la liste des paramètres pour le programme RPG en fonction des attributs contenus dans l'objet métier.
3. Il exécute le programme RPG correspondant à l'objet métier.
4. Il retourne le résultat de l'exécution du programme : réussite ou échec.

Pour créer des objets métier, vous utiliserez Business Object Designer ODA. Créez la définition de l'objet métier et ajoutez les attributs requis. Configurez ensuite le connecteur pour prendre en charge l'objet métier. Pour plus d'informations sur le Business Object Designer ODA, voir le Chapitre 5, «Création et modification d'objets métier», à la page 39.

Structure Business Object pour les files d'attente de données iSeries

Pour l'objet métier files d'attente de données, les attributs représentent les zones de la file d'attente de données. Il possède également un attribut Connection enfant. Il peut y avoir une relation de type parent-enfant si une AS400Structure est indiquée dans les zones de la file d'attente de données. Les verbes valides sont GETQUEUE et PUTQUEUE.

Remarque : Les noms de verbes ne sont pas sensibles à la casse. Par exemple, le verbe GetQueue peut s'écrire GETQUEUE ou GetQueue.

Les informations spécifiques à l'application pour les files d'attente figureront au niveau de l'objet métier. La valeur sera le chemin IFSFile absolu de la file d'attente de données. La longueur totale des attributs doit être égale à la longueur maximale d'un élément de la file d'attente. Cette valeur est définie lorsque vous créez la file d'attente sur la machine iSeries.

Le paramètre peut être de type Input, Output ou InOut. L'objet Connection et tous ses attributs sont définis de la façon requise. Par défaut, l'ODA iSeries génère tous les attributs avec ASI sous la forme `ParamType=InOut`. Il est toutefois possible de les faire passer sur Input ou Output après s'être assuré que la modification est conforme à la logique du programme.

Voici un exemple d'objet métier de file d'attente de données :

```
Verb=GETQUEUE
BO LEVEL ASI
QSYS.LIB/MYLIB.LIB/MYQUEUE.DTAQ

Connection Instance
        HostName=ibm.siberia.in.com
        UserName=Prapulla
        Password=Prapulla

Attribute1
```

Configuration des méta-objets en vue de l'interrogation

L'adaptateur iSeries utilise des méta-objets pour toutes les files d'attente de données qui reçoivent un message complet de file d'attente de données pour modification de la base de données ou des fichiers de données. Vous devez configurer les méta-objets de toutes les files d'attente de données qui reçoivent de tels messages.

Le nom du méta-objet commence toujours par `MO_iSeries`. Chaque méta-objet contient des informations sur les files d'attente de données. Vous devez ajouter un verbe factice à tous les méta-objets.

Les attributs (nom d'hôte, nom d'utilisateur et mot de passe) des méta-objets ont une valeur par défaut statique. Il n'est pas possible de modifier ces attributs par défaut de façon dynamique, une fois le connecteur démarré, puisque ces valeurs sont mises en mémoire cache dans l'agent du connecteur. Pour accéder à la même

file d'attente de données sur une machine différente, vous devez soit modifier la valeur par défaut et redémarrer l'instance de l'adaptateur iSeries, soit configurer un autre méta-objet pour les informations de la nouvelle machine.

Le verbe contenu dans l'objet métier est défini sur la chaîne appropriée, tandis que l'attribut DataQueueName est défini sur le chemin IFS File de la file d'attente de données. L'attribut BusObjName contient le nom de l'objet métier correspondant (qui contient le verbe mentionné dans le méta-objet). Les détails lus à partir de la file d'attente sont renseignés dans cet objet métier. Les attributs du méta-objet sont illustrés dans l'exemple ci-dessous (Figure 11). Le SamplePollBO correspondant est présenté dans la Figure 12.

MO_iSeries_Poll1:Local Project *

General | Attributes

	Name	Type	Key	Foreign Key	Requi red Attrib	Cardina lity	Maxim um Length	Default Value	Applic Spec Inform
1	UserName	String	☑	☐	☐		255	user	
2	PassWord	String	☐	☐	☐		255	passwd	
3	HostName	String	☐	☐	☐		255	iseries.server.com	
4	Verb	String	☐	☐	☐		255	Create	
5	DataQueueName	String	☐	☐	☐		255	/QSYS.LIB/PLIB.LIB/TESTDQ1.DTAQ	
6	BusObjName	String	☐	☐	☐		255	SamplePollBO	
7	ObjectEventId	String							
8			☐	☐	☐		255		

Figure 10. Exemple d'objet métier d'interrogation pour une file d'attente de données séquentielles

SamplePollBO:Local Project *

General | Attributes

	Pos	Name	Type	Key	Foreign Key	Requi red	Cardina lity	Maximu m	Default Value	Application Specific
1	1	TestField	String	☑	☐	☐		100		
2	2	ObjectEventId	String							
3	3			☐	☐	☐		255		

Figure 11. SamplePollBO correspondant

Pour les files d'attente de données indexées, un attribut de clé nommé "key" sera ajouté au méta-objet présenté à la Figure 13. L'appel d'interrogation utilisera les informations de la clé pour extraire le message correspondant de la file d'attente de données indexées.

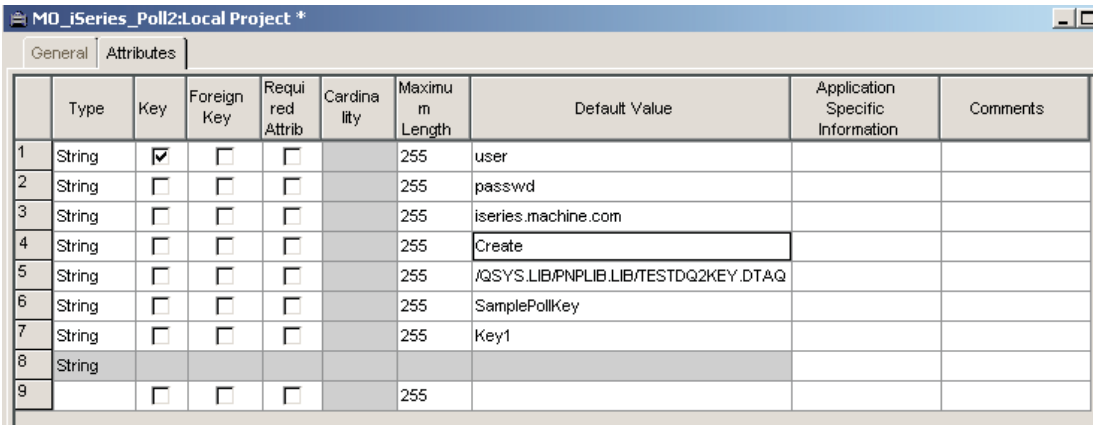


Figure 12. Exemple d'objet métier d'interrogation pour une file d'attente de données indexées

L'exemple d'objet métier d'interrogation correspondant est présenté dans la Figure 14.

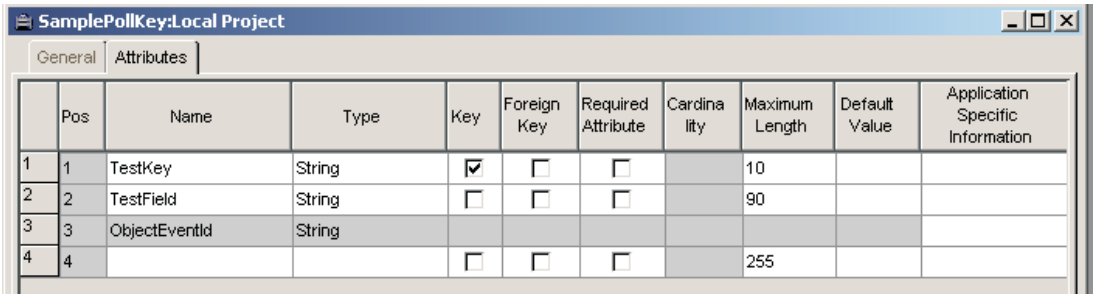


Figure 13. Exemple d'objet métier d'interrogation pour une file d'attente de données indexées

Spécification de propriétés d'attribut d'objet métier

Le connecteur iSeries dispose de plusieurs propriétés que vous pouvez définir dans ses attributs d'objet métier. Cette section indique comment le connecteur interprète quelques-unes de ces propriétés et décrit la manière de les définir lors de la modification d'un objet métier.

Le tableau ci-après illustre les propriétés des attributs simples.

Tableau 4. Propriétés d'attribut d'objets métier

Attribut	**Description**
Nom	Nom unique de l'attribut
Type	Tous les attributs simples doivent être de type String
MaxLength	Si la longueur de la valeur de l'attribut est supérieure à la valeur MaxLength précisée, et si l'attribut représente un paramètre input, la valeur est tronquée à la longueur Maxlength. Si la valeur est plus courte que MaxLength, des espaces sont ajoutés.
IsKey	Non utilisé
IsForeignKey	Non utilisé

Tableau 4. Propriétés d'attribut d'objets métier (suite)

Attribut	Description
Isrequired	Cet attribut doit être paramétré sur `true` pour tous les paramètres d'entrée.
AppSpecInfo	`ParamType=<value>:Offset=<value>: Signed=<True/False>:DataLength= <value>:PackedDec=<True/False>: ZonedDec=<True/False>: DecimalPositions=<value>`
DefaultValue	Si elle est définie pour l'attribut, cette valeur sera utilisée par le connecteur si aucune valeur n'est définie pour le paramètre d'entrée.

Spécification du texte de l'application au niveau de l'attribut de l'objet métier

Les informations suivantes figurent dans le texte de l'application au niveau de l'attribut de l'objet métier.

Tableau 5. Attributs de l'objet métier

Propriété	Valeurs	Description
`ParamType`	Input/Output/InOut	Indique le type de paramètre représenté par l'attribut.
`Offset`	Un nombre entier	Indique le décalage dans le tableau d'octets où commence la valeur du paramètre.
`Signed`	True/false	La propriété indique si les types de nombre integer/short/long sont signés. Si elle n'est pas définie, la valeur par défaut et non signé.
`DataLength`	Un nombre entier	S'applique aux types integer/short/long . Indique la longueur des données pour les types signés et non signés. Si elle n'est pas définie, la valeur par défaut est `4`.
`DecimalPositions`	Un nombre entier	S'applique aux types zoned decimal et packed decimal. Indique le nombre de positions décimales.
`PackedDec`	True/false	Défini sur `true`, l'attribut représente un décimal condensé.
`ZonedDec`	True/false	Défini sur `true`, l'attribut représente un décimal non condensé.

Conversion des données à partir du kit d'outils iSeries ou AS/400

Le kit d'outils pour iSeries/AS400 inclut des classes de conversion de données. Le tableau ci-dessous indique la correspondance entre les types de données iSeries/AS400 et les types de données IBM WebSphere Business Integration, ainsi que la classe de conversion de données utilisée.

Tableau 6. Types de données et classes de conversion

Type de données iSeries/AS400	**Type de données IBM WBI**	**Classe de conversion de données**
Nombre AS/400 sur deux octets, signé.	Integer - Info spécifique à l'application - Signed=true; DataLength=2	AS400Bin2
Nombre AS/400 sur quatre octets, signé.	Integer - Info spécifique à l'application - Signed=true; DataLength=4	AS400Bin4
Nombre AS/400 en virgule flottante, sur deux octets, signé.	Float	AS400Float4
Nombre AS/400 en virgule flottante, sur quatre octets, signé.	Double	AS400Float8
Nombre AS/400 sur deux octets, non signé.	Integer - Info spécifique à l'application - Signed=false; DataLength=2	AS400UnsignedBin2
Nombre AS/400 sur quatre octets, non signé.	Integer - Info spécifique à l'application - Signed=false; DataLength=4	AS400UnsignedBin4
Décimal condensé AS/400.	String - la propriété de l'attribut MaxLength doit avoir le nombre de chiffres. Info spécifique à l'application - DecimalPositions=<number of decimal positions>; PackedDec=true	AS400PackedDecimal
Décimal non condensé AS/400.	String - la propriété de l'attribut MaxLength doit avoir le nombre de chiffres. Info spécifique à l'application - DecimalPositions=<number of decimal positions>; ZonedDec=true	AS400ZonedDecimal
Données de type caractère	String - MaxLength indique la longueur maximale.	AS400Text
Données de date	String - MaxLength indique la longueur maximale.	AS400Text

Chapitre 5. Création et modification d'objets métier

Le présent chapitre décrit l'Object Discovery Agent (ODA) pour iSeries, ainsi que son utilisation pour générer des définitions d'objet métier pour IBM WebSphere Business Integration Adapter for iSeries.

Ce chapitre contient les sections suivantes :

- «Présentation de l'ODA pour iSeries»
- «Génération de définitions d'objets métier»
- «Indication des informations sur l'objet métier», à la page 45
- «Téléchargement d'objets métier», à la page 47

Présentation de l'ODA pour iSeries

Un Object Discovery Agent (ODA) permet de générer des définitions d'objet métier. Une définition d'objet métier est un modèle d'objet métier. L'ODA examine les objets d'application spécifiés, "reconnaît" les éléments de ces objets qui correspondent aux attributs d'objet métier, et génère des définitions d'objet métier pour représenter les informations. Business Object Designer fournit une interface graphique pour accéder à l'Object Discovery Agent et l'utiliser de façon interactive.

L'ODA pour iSeries génère des définitions d'objet métier permettant d'accéder à des programmes RPG et RPGLE ainsi qu'à des objets de file d'attente de données sur système iSeries. L'assistant Business Object Designer automatise la création de ces définitions. Vous utiliserez l'ODA pour créer des objets métier, et Connector Configurator pour configurer le connecteur nécessaire pour les prendre en charge.

Génération de définitions d'objets métier

Cette section explique comment utiliser iSeries ODA dans Business Object Designer pour créer des définitions d'objet métier. Pour plus d'informations sur le lancement et l'utilisation de Business Object Designer, voir *IBM WebSphere Business Integration Adapters Business Object Development Guide*.

Démarrage de l'ODA iSeries

Vous pouvez démarrer l'ODA iSeries à l'aide d'un des scripts suivants :

- Sous Windows – `start_iSeriesODA.bat`

  Remarque : Vous pouvez également démarrer l'ODA iSeries à l'aide du raccourci créé automatiquement par le programme d'installation dans les environnements Windows.

- Pour UNIX – `start_iSeriesODA.sh`

Utilisez Business Object Designer pour sélectionner, configurer et exécuter l'ODA iSeries. Business Object Designer recherche chaque ODA par le nom indiqué dans la variable `AGENTNAME` de chaque script ou fichier de traitement par lots.

Exécution de Business Object Designer

Business Object Designer dispose d'un assistant pour vous guider à travers les étapes nécessaires pour générer une définition d'objet métier en utilisant l'ODA.

Sélection de l'agent

Vous devez d'abord sélectionner l'agent ODA.

1. Ouvrez Business Object Designer.
2. Cliquez sur **File** > **New Using ODA**. La fenêtre Business Object Wizard - Step 1 of 6 - Select Agent s'ouvre.
3. Sélectionnez l'ODA/AGENTNAME (du script start_iSeriesODA) dans la liste Located agents et cliquez sur Next. (Vous devrez peut-être cliquer sur Find Agents si l'agent souhaité n'est pas répertorié.)

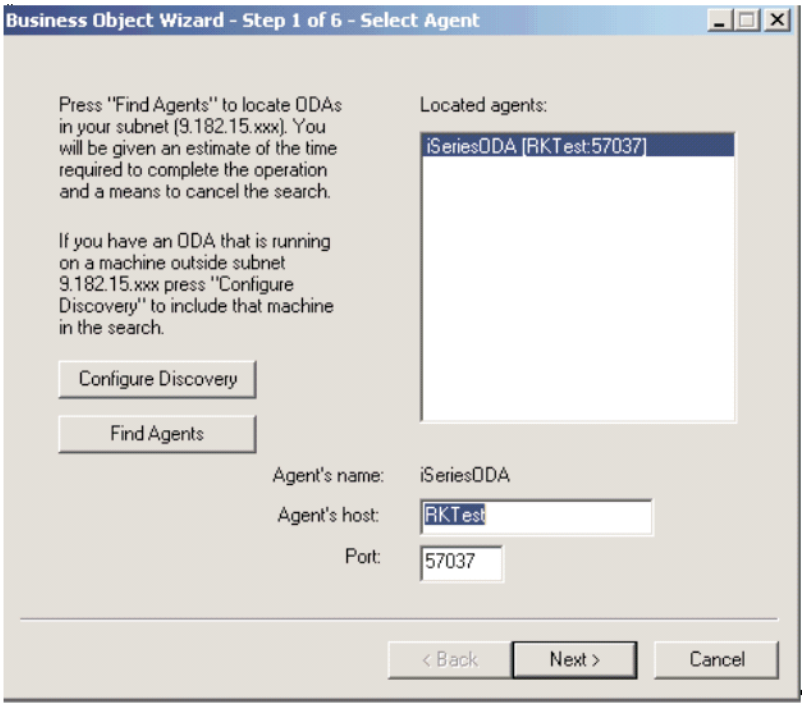


Figure 14. Fenêtre Select Agent

Configuration de l'agent

Une fois que vous avez cliqué sur Next dans la fenêtre Select Agent, la fenêtre Business Object Wizard - Step 2 of 6 - Configure Agent s'ouvre.

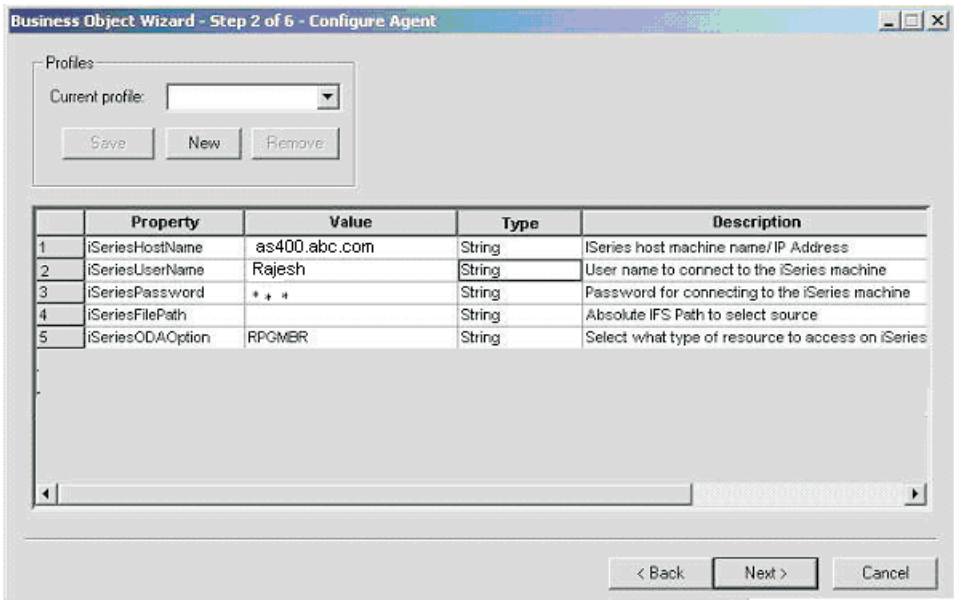


Figure 15. Fenêtre Configure Agent

Les propriétés définies dans cet écran sont décrites dans le Tableau 5. Vous pouvez enregistrer dans un profil toutes les valeurs entrées dans cet écran. Au lieu de saisir de nouveau les données de la propriété la prochaine fois que vous exécuterez l'ODA, vous sélectionnerez le profil voulu dans le menu déroulant et ré-utilisez les valeurs enregistrées. Vous pouvez enregistrer plusieurs profils, chacun avec un ensemble de valeurs différent.

Tableau 7. Propriétés de configuration de l'agent

Nom de propriété	Valeur par défaut	Type	Description
iSeriesHostName		String	(obligatoire) Nom de la machine hôte iSeries.
iSeriesUserName		String	(obligatoire) Nom d'utilisateur requis pour se connecter à la machine iSeries
iSeriesPassword		String	(obligatoire) Mot de passe utilisé pour se connecter à la machine iSeries
iSeriesFilePath	/QSYS.LIB/	String	Chemin IFS absolu vers la source de sélection.

Tableau 7. Propriétés de configuration de l'agent (suite)

Nom de propriété	Valeur par défaut	Type	Description
iSeriesODAOption		String	Type de ressource utilisé par le iSeries. Actuellement, il y a trois options : RPGMBR, RPGLEMBR et DTAQ. Si RPGMBR est sélectionné, les IFSFiles entrés sont considérés comme des programmes sources au format RPG. Si RPGLE est entré, les IFSFiles sélectionnés sont considérés comme des programmes sources au format RPGLE. DTAQ implique que le fichier IFS sélectionné est un DataQueue, qui peut être séquentiel ou indexé. En fonction de cette sélection, l'accès aux fichiers sources permet de créer les définitions d'objet métier correspondantes.

Remarque : Des programmes sources RPG et RPGLE de différentes longueurs d'enregistrement sont pris en charge, en fonction des définitions de format du langage.

Utilisez les boutons New et Save de la zone de groupe Profiles dès que vous voulez que l'ODA crée un nouveau profil. Lorsque vous utiliserez de nouveau l'ODA, vous pouvez sélectionner un profil existant. Saisissez la valeur de chaque propriété, telle que définie dans le tableau 7, à la page 41.

Lorsqu'une zone obligatoire est laissée vide ou en cas d'erreur (par exemple si un nom d'utilisateur est incorrect), le message d'erreur correspondant s'affiche en incrustation.

Remarque : Si vous utilisez un profil, les valeurs de la propriété sont remplies automatiquement, mais vous pouvez les modifier si nécessaire. Vous pouvez également enregistrer de nouvelles valeurs.

Sélection d'un objet métier

La fenêtre Business Object Wizard - Step 3 of 6 - Select Source s'ouvre, comme illustré sur la Figure 17.

Cet écran dresse la liste des fichiers sources *.MBRs pour RPG ou RPGLE ou des fichiers *.DTAQ pour files d'attente de données, pour que l'utilisateur sélectionne les noms de ces fichiers. Le type de fichier est déterminé par la propriété iSeriesODAOption Agent. Les répertoires IFS sont représentés en tant que noeuds d'arborescence développables, tandis que les noms de sources (MBR ou DTAQ) sont présentés en tant que noeuds feuille. Vous pouvez sélectionner plusieurs sources (noeuds feuille uniquement) du même répertoire IFS ou de différents répertoires IFS. Utilisez cet écran pour sélectionner n'importe le nombre voulu de fichiers sources pour lesquels l'ODA générera des définitions d'objet métier.

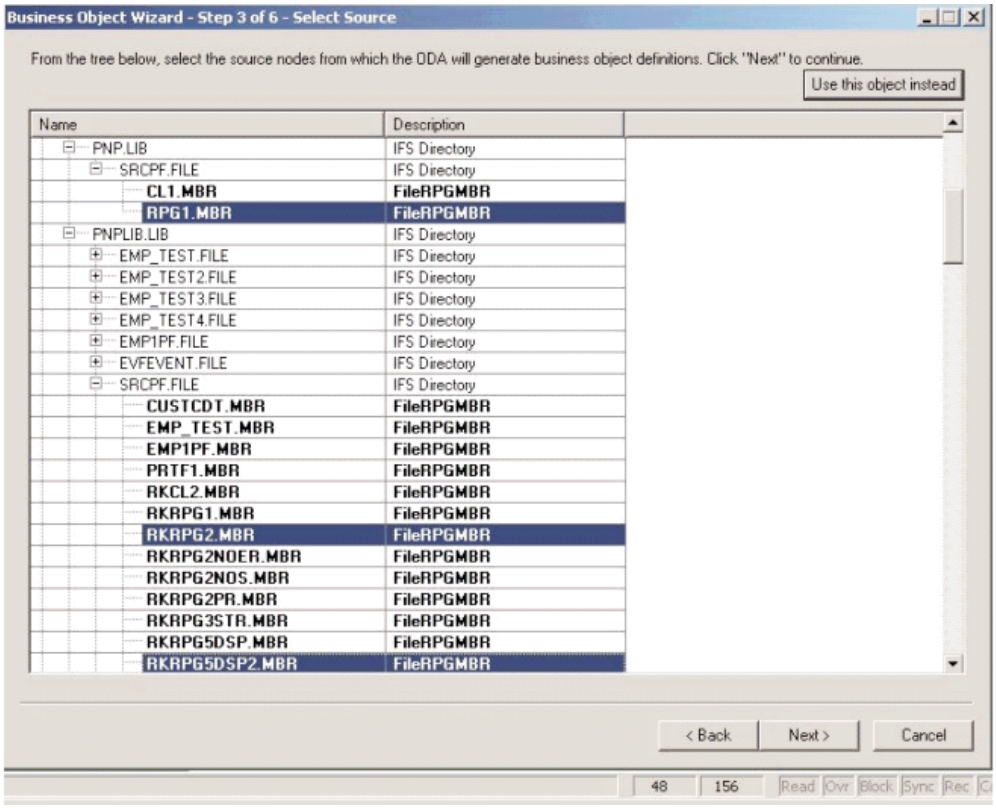


Figure 16. Fenêtre Select Source

1. Si nécessaire, développez un noeud pour consulter la liste des noeuds secondaires.
2. Sélectionnez les fichiers sources à utiliser. Cliquez sur Next.
3. Pour sélectionner plusieurs noeuds, voir *Business Object Development Guide* pour plus d'informations sur les structures d'arborescences.

Confirmation de la sélection d'objet

La fenêtre Business Object Wizard - Step 4 of 6 - Confirm source nodes for business object definitions s'ouvre. Elle affiche les objets sélectionnés.

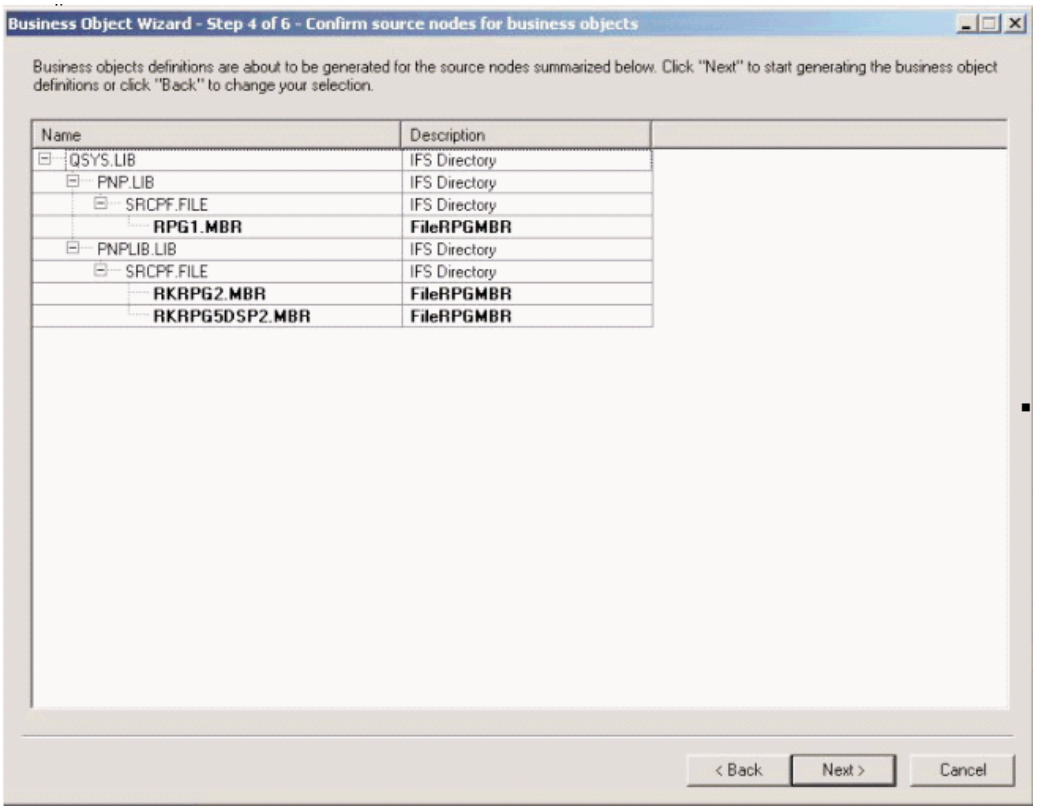


Figure 17. Fenêtre Confirm source node

Cliquez sur Back pour procéder à des modifications ou sur Next pour confirmer que la liste est correcte. La fenêtre Business Object Wizard - Step 5 of 6 - Generating business objects... s'ouvre avec un message indiquant que l'assistant est en train de générer les objets métier.

Génération d'objets métier

Une fois que vous avez confirmé vos sources de noeud, l'ODA iSeries génère les objets métier. La fenêtre Business Object Wizard - Step 6 of 6 - Saving business object definitions... s'ouvre.

1. Cochez soit Save a copy of the business object definitions to a separate file, soit Open the new business object definitions in separate windows. Ce dernier choix lance Business Object Designer et ouvre les objets métier dans cette application.
2. Si vous avez terminé et souhaitez fermer l'ODA, cochez Shutdown ODA et cliquez sur Finish.

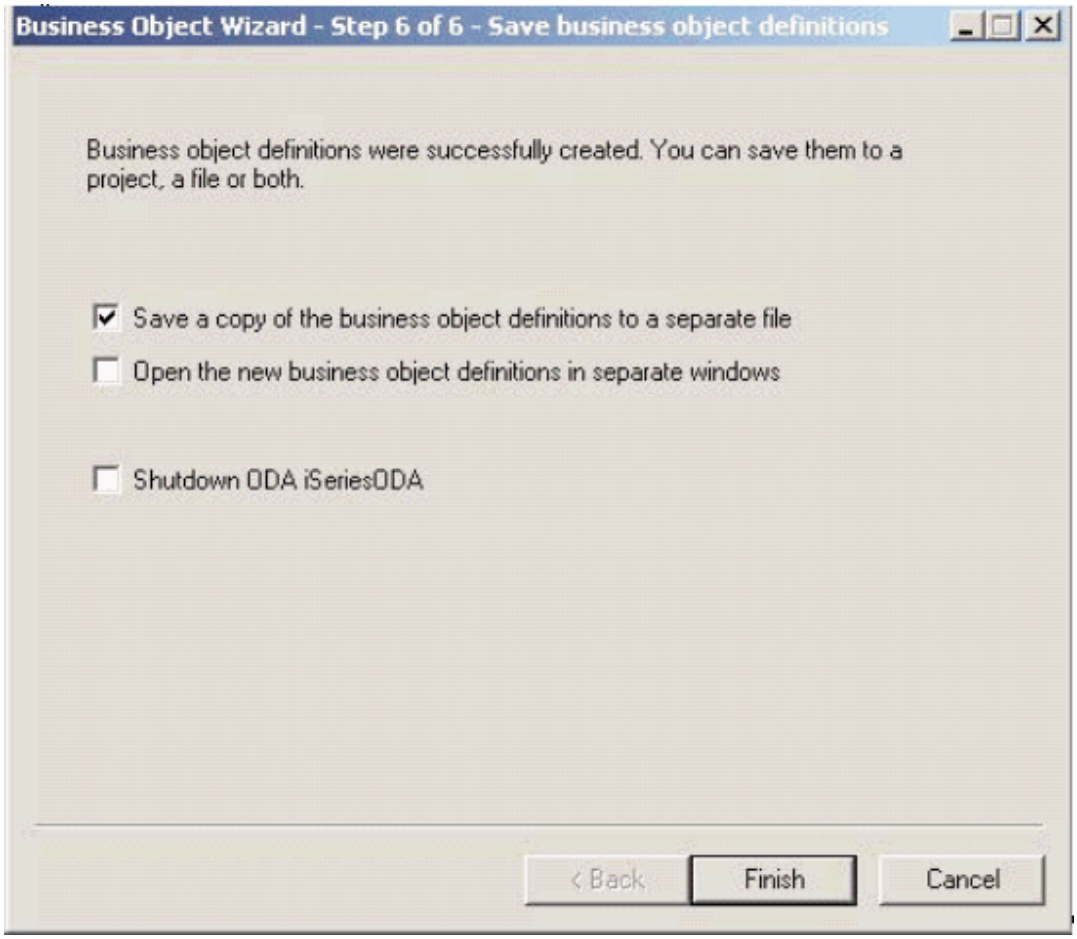


Figure 18. Fenêtre Save business objects

Indication des informations sur l'objet métier

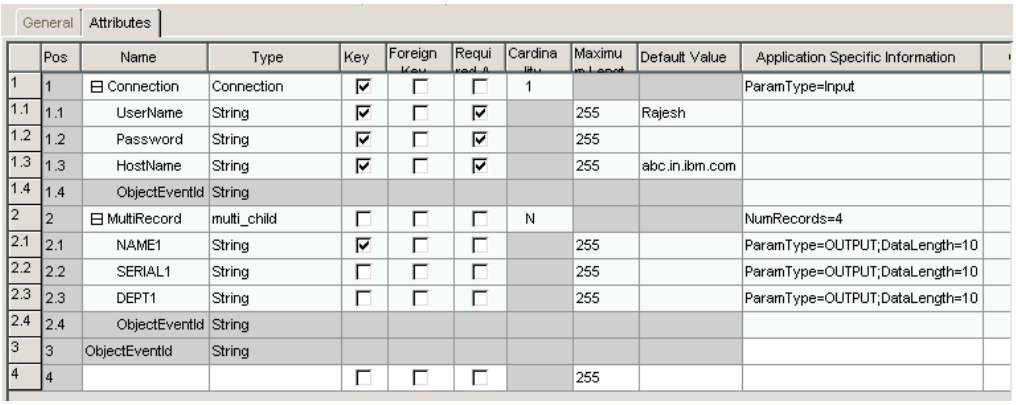
Une fois que vous avez créé un objet métier, vous pouvez préciser son ASI et celui de l'attribut.

Cette section indique comment spécifier ces informations à l'aide de l'ODA avec Business Object Designer. Pour une description détaillée de ces catégories d'informations et leur signification pour la structure d'objet métier du connecteur iSeries, voir le Chapitre 4, «Présentation des objets métier pour le connecteur», à la page 29.

Indication de l'ASI de l'attribut

Business Object Designer affiche les attributs de l'objet métier. Pour plus d'informations sur l'ASI de l'attribut du connecteur iSeries, voir «Spécification du texte de l'application au niveau de l'attribut de l'objet métier», à la page 37.

Les attributs sont indiqués dans l'onglet Attributes, dans l'ordre dans lequel ils apparaissent dans la structure de l'objet métier, tel que défini par la valeur numérique de la colonne Pos.



General | Attributes

	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	⊟ Connection	Connection	☑	☐	☐	1			ParamType=Input
1.1	1.1	UserName	String	☑	☐	☑		255	Rajesh	
1.2	1.2	Password	String	☑	☐	☑		255		
1.3	1.3	HostName	String	☑	☐	☑		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	⊟ MultiRecord	multi_child	☐	☐	☐	N			NumRecords=4
2.1	2.1	NAME1	String	☑	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

Figure 19. Définition de l'ASI d'attribut

La fenêtre indique le nom, le type et les informations ASI de chaque attribut. Dans cette fenêtre, vous devez préciser une clé (exigée par Business Object Designer pour valider et enregistrer un objet métier) pour chaque objet métier pour lequel l'ODA n'a pas encore indiqué de clé.

Dans les objets métier, les mots de passe ne sont pas définis comme valeur par défaut et ne font pas l'objet d'une trace, pour des raisons de sécurité. L'objet métier généré pour une file d'attente de données séquentielles possède un attribut correspondant à la longueur de ses données. Les files d'attente de données indexées possèdent deux attributs, le premier correspond à la clé et le second au reste de la longueur, c'est-à-dire Data Length - Key Length. Dans le cas d'un objet métier possédant un attribut dont la longueur de données ASI est égale à X et le nombre de positions décimales égal à Y, et si un BO est envoyé avec une valeur d'attribut supérieure à X chiffres (le nombre de chiffres en positions décimales étant supérieur à Y), le connecteur tronquera cette valeur à une longueur de partie décimale égale à Y, maintiendra la longueur des données à X et fera aboutir le traitement de l'objet métier. Si la partie entière dépasse la longueur X-Y, alors le connecteur émet une erreur. Par exemple, si l'ASI d'un attribut est PackedDecimal=True;Datalength=10;DecimalPositions=2, alors les valeurs 112345678, 12345678.1, et 12345678.12 sont acceptées, mais les chiffres décimaux situés après la limite maximale de 2 dans 12345678.123 seront tronqués et la valeur ramenée à 12345678.12. Le message de trace correspondant est ″Truncated String Value:<12345678.12> for DecimalPositions=2″. La valeur 123456789.12 consignera une erreur :″Length is not valid.″

Vous pouvez également utiliser cette fenêtre pour définir des clés d'objets enfants si nécessaire et préciser les informations suivantes :

- L'attribut est-il obligatoire pour que le connecteur traite l'objet métier ? Si c'est le cas, cochez la case Required.
- La longueur maximale de l'attribut est-elle différente de la valeur affichée dans la colonne Maximum Length ?
- L'attribut a-t-il une valeur par défaut ? Si c'est le cas, saisissez la valeur indiquée dans la colonne Default.

Indication de l'ASI de l'objet métier

Une fois que vous avez précisé l'ASI au niveau de l'attribut, vous pouvez afficher et modifier l'ASI au niveau de l'objet métier. Pour plus d'informations sur l'ASI de l'objet métier, voir «Spécification du texte de l'application au niveau de l'attribut de l'objet métier», à la page 37.

L'ASI de l'objet métier est indiqué dans l'onglet General. La valeur ASI indiquée dans les informations spécifiques de la zone Business Object Level contient le nom de la classe proxy représentant cet objet métier. Le connecteur utilise ces informations pour mapper une classe proxy sur un objet métier, et dans le cas d'un objet métier de serveur (lorsque le connecteur s'exécute également en tant que serveur), le connecteur utilise ces informations pour mapper une classe d'implémentation sur un objet métier.

Cet écran indique également tous les verbes pris en charge par l'objet métier et l'ASI de chaque verbe. Dans cet écran, vous pouvez modifier l'ASI de l'objet métier et les verbes pris en charge.

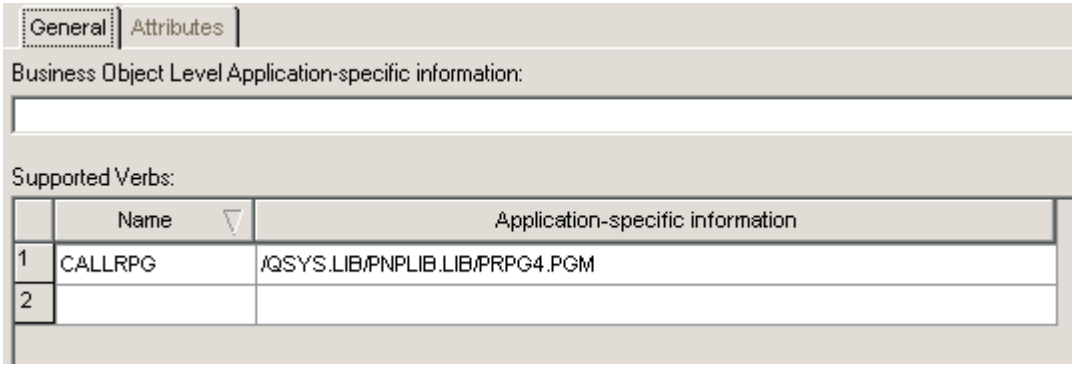


Figure 20. Configuration de l'ASI de l'objet métier

Téléchargement d'objets métier

Les fichiers de définition d'objets métier nouvellement créés doivent être téléchargés dans le courtier d'intégration. Le processus de téléchargement varie selon que vous exécutez WebSphere InterChange Server, WebSphere MQ Integrator Broker ou WebSphere Application Server :

- WebSphere InterChange Server : Si vous avez enregistré vos fichiers de définition d'objet métier sur une machine locale et devez les télécharger dans le référentiel du serveur, consultez *Implementation Guide for WebSphere InterChange Server*.
- WebSphere MQ Integrator Broker : Vous devez exporter les définitions d'objets métier depuis Business Object Designer dans le courtier d'intégration. Pour plus d'informations, voir *Implementing Adapters with WebSphere MQ Integrator Broker*.
- WebSphere Application Server : Pour plus d'informations, voir *Implementing Adapters with WebSphere Application Server*.

Chapitre 6. Identification et résolution des erreurs

Le présent chapitre décrit comment l'adaptateur pour iSeries gère les erreurs. L'adaptateur génère des messages de consignation et de traçage.

Ce chapitre contient les sections suivantes :

- «Gestion des erreurs»
- «Consignation»
- «Messages de traçage»

Gestion des erreurs

Tous les messages d'erreur générés par le connecteur sont conservés dans un fichier de messages nommé `BIA_ISERIESAdapter.txt`. (Le nom du fichier est déterminé par la propriété standard de configuration du connecteur `LogFileName`.)

Toutes les erreurs se traduisent par `VerbProcessingFailedException`.

Consignation

L'adaptateur consigne un message d'erreur dès qu'il rencontre un état anormal en cours de traitement, quel que soit le niveau de trace. Lorsqu'une erreur survient, le connecteur imprime également une représentation textuelle de l'objet métier qui a échoué tel qu'il a été reçu. Il écrit le texte dans le fichier journal de l'adaptateur iSeries, dont le nom de fichier correspond à la propriété de connecteur LogFileName. Le message contient une description détaillée de la condition et du résultat et peut également inclure des informations supplémentaires pouvant servir au débogage, telles que les suppressions d'objets métier et traces de pile (pour les exceptions).

Les ID des messages de l'adaptateur iSeries vont de 93000 à 94000.

Messages de traçage

Le traçage est une fonction de débogage facultative que vous pouvez activer pour suivre de près le comportement d'un connecteur. Les messages de traçage sont configurables et peuvent être modifiés de manière dynamique. Vous pouvez définir divers niveaux en fonction du niveau de détail souhaité. Par défaut, les messages de trace sont consignés dans `STDOUT`.

Vous pouvez également configurer le traçage de sorte qu'il soit consigné dans un fichier. Le tableau suivant décrit les types de messages de trace émis par le connecteur iSeries pour chaque niveau de trace. Tous les messages de trace apparaissent dans le fichier spécifié par la propriété TraceFileName du connecteur. Ces messages viennent compléter les messages de traçage générés par l'architecture IBM WebSphere Business Integration Adapter. Pour plus d'informations sur la configuration des messages de trace, voir les propriétés de configuration du connecteur dans "Configuration du connecteur." Pour plus d'informations sur le traçage, en particulier pour savoir comment l'activer et le paramétrer, voir Connector Development Guide.

Le Tableau 6 dresse la liste du contenu recommandé pour les niveaux de message de traçage du connecteur.

Tableau 8. Description du niveau de contenu des messages de trace

Niveau de trace	Messages de traçage
Niveau 0	Utilisez ce niveau de trace pour les messages de trace qui identifient la version du connecteur. Aucun autre traçage n'est réalisé à ce niveau.
Niveau 1	Non applicable
Niveau 2	Utilisez ce niveau de trace pour les messages de trace qui : • Identifient le gestionnaire de BO utilisé pour chaque objet traité par le connecteur. • Signalent chaque fois qu'un objet métier est transmis au courtier d'intégration • Signalent chaque fois qu'un objet métier de requête est reçu
Niveau 3	Non applicable
Niveau 4	Utilisez ce niveau de trace pour les messages de trace qui : • Identifient les informations spécifiques à l'application. C'est le cas lorsque des valeurs sont retournées par les méthodes qui traitent les zones d'information spécifiques à l'application dans les objets métier. • Identifient lorsque le connecteur entre ou quitte une fonction. Ces messages aident à procéder au traçage du flot de processus du connecteur. • Enregistrent tout traitement spécifique à l'unité d'exécution. Par exemple, si le connecteur engendre plusieurs unités d'exécution, un message signale la création de chacune.

Tableau 8. Description du niveau de contenu des messages de trace (suite)

Niveau de trace	Messages de traçage
Niveau 5	Utilisez ce niveau de trace pour les messages de trace qui : • Indiquent l'initialisation du connecteur. Ce type de message peut inclure par exemple la valeur de chaque propriété de configurateur de connecteur, extraite du courtier. • Détaillent l'état de chaque unité d'exécution que le connecteur engendre pendant son exécution. • Représentent des instructions exécutées dans l'application. Le fichier de consignation du connecteur contient toutes les instructions exécutées dans l'application cible ainsi que la valeur des variables qui sont remplacées. • Enregistrent les suppressions d'objets métier. Le connecteur sort une représentation textuelle d'un objet métier avant le début du traitement (indiquant l'objet que le connecteur reçoit de la collaboration), et après la fin du traitement de l'objet (indiquant l'objet retourné par le connecteur à la collaboration).

Annexe. Propriétés de configuration standard pour les connecteurs

Cette annexe décrit les propriétés de configuration standard pour le composant de connecteur de WebSphere Business Integration Adapters. Les informations présentées concernent les connecteurs qui s'exécutent sur les courtiers d'intégration suivants :

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator, WebSphere MQ Integrator Broker et WebSphere Business Integration Message Broker, appelés collectivement courtiers de messages WebSphere (et WMQI dans Connector Configurator).
- Information Integrator (II)
- WebSphere Application Server (WAS)

Si votre adaptateur prend en charge DB2 Information Integrator, utilisez les options et les propriétés standard DB2 II (voir la colonne Remarques du tableau 9, à la page 55.)

Les propriétés définies pour l'adaptateur dépendent du courtier d'intégration utilisé. Vous sélectionnez ce dernier à l'aide de Connector Configurator. Une fois le courtier choisi, Connector Configurator dresse la liste des propriétés standard à configurer pour l'adaptateur.

Pour plus d'informations sur les propriétés spécifiques au connecteur, voir la section correspondante de ce guide.

Nouvelles propriétés

La propriété standard suivante a été ajoutée à cette édition :

- BOTrace

Présentation des propriétés de connecteur standard

Les connecteurs ont deux types de propriétés de configuration :

- Les propriétés de configuration standard, utilisées par l'architecture
- Les propriétés de configuration spécifiques à une application ou à un connecteur, et utilisées par l'agent

Ces propriétés déterminent l'architecture de l'adaptateur et le comportement d'exécution de l'agent.

Cette section indique comment démarrer Connector Configurator et décrit les caractéristiques communes à toutes les propriétés. Pour plus d'informations sur les propriétés de configuration spécifiques à un connecteur, reportez-vous au guide d'utilisateur de l'adaptateur approprié.

Démarrage de Connector Configurator

Vous pouvez configurer les propriétés du connecteur à partir de Connector Configurator, accessible via System Manager. Pour plus d'informations sur l'utilisation de Connector Configurator, voir les sections associées de ce guide.

Connector Configurator et System Manager s'exécutent uniquement sous Windows. Si vous exécutez le connecteur sous UNIX, vous devez posséder une machine Windows sur laquelle ces outils sont installés.

Pour définir les propriétés d'un connecteur s'exécutant sous UNIX, vous devez démarrer System Manager sur la machine Windows, établir une connexion au courtier d'intégration UNIX et mettre à jour Connector Configurator pour le connecteur.

Présentation des valeurs des propriétés de configuration

Le connecteur utilise l'ordre suivant pour déterminer la valeur d'une propriété :

1. Valeur par défaut
2. Référentiel (valide uniquement si WebSphere InterChange Server (ICS) est le courtier d'intégration)
3. Fichier de configuration locale
4. Ligne de commande

La longueur par défaut d'une propriété est de 255 caractères. La longueur d'un type de propriété STRING n'est pas limitée. La longueur d'un type INTEGER est déterminée par le serveur sur lequel l'adaptateur fonctionne.

Un connecteur obtient ses valeurs de configuration lors du démarrage. Si vous modifiez la valeur d'une ou plusieurs propriétés du connecteur pendant une session d'exécution, la méthode de mise à jour de la propriété détermine la manière dont les modifications prennent effet.

Les caractéristiques de mise à jour d'une propriété, c'est-à-dire à quel moment et de quelle façon la modification des propriétés du connecteur prend effet, dépendent de la nature de la propriété.

Il existe quatre méthodes de mise à jour pour les propriétés standard du connecteur :

- **Dynamique**
  La nouvelle valeur prend effet dès que la modification est enregistrée dans System Manager. Toutefois, si le connecteur est en mode autonome (indépendamment de System Manager), par exemple avec l'un des courtiers de message WebSphere, vous ne pouvez modifier les propriétés que via le fichier de configuration. Dans ce cas, une mise à jour dynamique n'est pas possible.
- **Redémarrage de l'agent (ICS uniquement)**
  La nouvelle valeur ne prend effet qu'une fois que vous avez arrêté et redémarré l'agent du connecteur.
- **Redémarrage du composant**
  La nouvelle valeur ne prend effet qu'après que le connecteur ait été arrêté et redémarré dans System Manager. Vous n'avez pas besoin d'arrêter et de redémarrer l'agent ni le processus du serveur.
- **Redémarrage du système**
  La nouvelle valeur ne prend effet qu'une fois que vous avez arrêté et redémarré l'agent du connecteur et le serveur.

Pour déterminer la manière dont une propriété donnée est mise à jour, reportez-vous à la colonne **Update Method**µ dans la fenêtre Connector Configurator, ou à la colonne Update Method dans le tableau 9, à la page 55.

Une propriété standard peut figurer à trois endroits. Certaines propriétés peuvent figurer à plusieurs emplacements.

- **ReposController**
  La propriété réside dans le contrôleur du connecteur et n'est effective qu'à cet emplacement. Le fait de modifier la valeur du côté de l'agent n'a pas d'effet sur le contrôleur.
- **ReposAgent**
  La propriété réside dans l'agent et n'est effective qu'à cet emplacement. Selon la propriété, une configuration locale peut remplacer cette valeur.
- **LocalConfig**
  La propriété réside dans le fichier de configuration du connecteur et n'agit que par l'intermédiaire de ce fichier. Le contrôleur ne peut pas modifier la valeur de la propriété et n'est pas informé des modifications apportées au fichier de configuration, à moins que le système ne soit redéployé pour remettre à jour le contrôleur explicitement.

Référence rapide des propriétés standard

Le tableau 9 fournit une description rapide des propriétés standard de configuration des connecteurs. Les connecteurs n'exigent pas tous toutes ces propriétés, et les paramètres de propriété peuvent différer d'un courtier d'intégration à l'autre.

Voir la section qui suit le tableau pour une description de chaque propriété.

Remarque : Dans la colonne "Remarques" de tableau 9, la phrase "La valeur de RepositoryDirectory est égale à <REMOTE>" indique que le courtier est InterChange Server. Lorsque le courtier est WMQI ou WAS, le répertoire du référentiel est défini sur <*ProductDir*>\repository

Tableau 9. Récapitulatif des propriétés de configuration standard

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
AdapterHelpName	Un des sous-répertoires valides de <*ProductDir*>\bin\Data \App\Help\ contenant un répertoire <*RegionalSetting*> valide	Nom du modèle, si valide, ou zone vide	Redémarrage du composant	Paramètres régionaux pris en charge. Incluent `chs_chn`, `cht_twn`, `deu_deu`, `esn_esp`, `fra_fra`, `ita_ita`, `jpn_jpn`, `kor_kor`, `ptb_bra`, et `enu_usa` (par défaut).
AdminInQueue	Nom de file d'attente JMS valide	<*CONNECTORNAME*> / `ADMININQUEUE`	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est `JMS`
AdminOutQueue	Nom de file d'attente JMS valide	<*CONNECTORNAME*>/ `ADMINOUTQUEUE`	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est `JMS`

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
AgentConnections	1 à 4	1	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est MQ ou IDL, la valeur de Repository Directory est <REMOTE> et la valeur de BrokerType est ICS.
AgentTraceLevel	0 à 5	0	Dynamique si le courtier est ICS ; sinon Redémarrage du composant	
ApplicationName	Nom d'application	Valeur précisée pour le nom de l'application du connecteur	Redémarrage du composant	
BiDi.Application	Toute combinaison valide de ces attributs bidirectionnels : 1ère lettre : I,V 2e lettre : L,R 3e lettre : Y, N 4e lettre : S, N 5e lettre : H, C, N	ILYNN (cinq lettres)	Redémarrage du composant	Cette propriété n'est valide que si la valeur de BiDi.Transforma tion est true
BiDi.Broker	Toute combinaison valide de ces attributs bidirectionnels : 1ère lettre : I,V 2e lettre : L,R 3e lettre : Y, N 4e lettre : S, N 5e lettre : H, C, N	ILYNN (cinq lettres)	Redémarrage du composant	Cette propriété n'est valide que si la valeur de BiDi.Transformation est true. Si la valeur de BrokerType est ICS, la propriété est en lecture seule.
BiDi.Metadata	Toute combinaison valide de ces attributs bidirectionnels : 1ère lettre : I,V 2e lettre : L,R 3e lettre : Y, N 4e lettre : S, N 5e lettre : H, C, N	ILYNN (cinq lettres)	Redémarrage du composant	Cette propriété n'est valide que si la valeur de BiDi.Transformation est true.
BiDi.Transformation	true ou false	false	Redémarrage du composant	Cette propriété n'est valide que si la valeur de BrokerType n'est pas WAS.
BOTrace	none ou keys ou full	none	Redémarrage du composant	Cette propriété n'est valide que si la valeur de AgentTraceLevel est inférieure à 5.
BrokerType	ICS , WMQI, WAS	ICS	Redémarrage du composant	

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
CharacterEncoding	Tout code pris en charge. La liste indique le sous-ensemble : `ascii7`, `ascii8`, `SJIS`, `Cp949`, `GBK`, `Big5`, `Cp297`, `Cp273`, `Cp280`, `Cp284`, `Cp037`, `Cp437`.	`ascii7`	Redémarrage du composant	Cette propriété n'est valide que pour les connecteurs C++.
CommonEventInfrastructure	`true` ou `false`	`false`	Redémarrage du composant	
CommonEventInfrastructureURL	Une chaîne URL, par exemple, `corbaloc:iiop:host:2809`.	Aucune valeur par défaut.	Redémarrage du composant	Cette propriété n'est valide que si la valeur de CommonEvent Infrastructure est `true`.
ConcurrentEventTriggeredFlows	`1` à `32,767`	`1`	Redémarrage du composant	Cette propriété n'est valide que si la valeur de RepositoryDirectory est <REMOTE> et la valeur de BrokerType est ICS.
ContainerManagedEvents	Vide ou `JMS`	Vide	Redémarrage du composant	Cette propriété n'est valide que si la valeur de Delivery Transport est `JMS`.
ControllerEventSequencing	`true` ou `false`	`true`	Dynamique	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
ControllerStoreAndForwardMode	`true` ou `false`	`true`	Dynamique	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
ControllerTraceLevel	`0` à `5`	`0`	Dynamique	Cette propriété n'est valide que si la valeur de RepositoryDirectory est <REMOTE> et la valeur de BrokerType est ICS.
DeliveryQueue	Tout nom valide de file d'attente JMS valide	*<CONNECTORNAME>* `/DELIVERYQUEUE`	Redémarrage du composant	Cette propriété n'est valide que si la valeur de Delivery Transport est `JMS`.
DeliveryTransport	`MQ`, `IDL` ou `JMS`	`IDL`lorsque la valeur de RepositoryDirectory est <REMOTE>, sinon `JMS`	Redémarrage du composant	Si la valeur de RepositoryDirectory n'est pas <REMOTE>, la seule valeur valide pour cette propriété est `JMS`.

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
DuplicateEventElimination	true ou false	false	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
EnableOidForFlowMonitoring	true ou false	false	Redémarrage du composant	Cette propriété n'est valide que si la valeur de BrokerType est ICS.
FaultQueue	Tout nom de file d'attente valide.	*<CONNECTORNAME>* /FAULTQUEUE	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
jms.FactoryClassName	CxCommon.Messaging.jms .IBMMQSeriesFactory, CxCommon.Messaging .jms.SonicMQFactory, ou n'importe quel nom de classe Java	CxCommon.Messaging. jms.IBMMQSeriesFactory	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
jms.ListenerConcurrency	1 à 32767	1	Redémarrage du composant	Cette propriété n'est valide que si la valeur de jms.TransportOptimized est true.
jms.MessageBrokerName	Si la valeur de jms.FactoryClassName est IBM, utilisez crossworlds.queue. manager.	crossworlds.queue. manager	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
jms.NumConcurrent Requests	Entier positif	10	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
jms.Password	Tout mot de passe valide		Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
jms.TransportOptimized	true ou false	false	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS et la valeur de BrokerType est ICS.
jms.UserName	Tout nom valide		Redémarrage du composant	Cette propriété n'est valide que si la valeur de Delivery Transport est JMS.
JvmMaxHeapSize	Taille de segment en mégaoctets	128 Mo	Redémarrage du composant	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
JvmMaxNativeStackSize	Taille de la pile en kilo-octets	128 Ko	Redémarrage du composant	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
JvmMinHeapSize	Taille de segment en mégaoctets	1 Mo	Redémarrage du composant	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
ListenerConcurrency	1 à 100	1	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est MQ.
Locale	Il s'agit d'un sous-ensemble des paramètres régionaux pris en charge : en_US, ja_JP, ko_KR, zh_CN, zh_TW, fr_FR, de_DE, it_IT, es_ES, pt_BR	en_US	Redémarrage du composant	
LogAtInterchangeEnd	true ou false	false	Redémarrage du composant	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
MaxEventCapacity	1 à 2147483647	2147483647	Dynamique	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
MessageFileName	Nom de fichier valide	InterchangeSystem.txt	Redémarrage du composant	
MonitorQueue	Tout nom de file d'attente valide	<*CONNECTORNAME*>/ MONITORQUEUE	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DuplicateEventElimination est true et si ContainerManaged-Events n'a pas de valeur.
OADAutoRestartAgent	true ou false	false	Dynamique	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
OADMaxNumRetry	Un nombre entier positif	1000	Dynamique	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
OADRetryTimeInterval	Un nombre entier positif en minutes	10	Dynamique	Cette propriété n'est valide que si la valeur de Repository Directory est égale à <REMOTE> et la valeur de BrokerType est ICS.
PollEndTime	HH = 0 à 23 MM = 0 à 59	HH:MM	Redémarrage du composant	
PollFrequency	Un nombre entier positif (en millisecondes)	10000	Dynamique si le courtier est ICS ; sinon Redémarrage du composant	
PollQuantity	1 à 500	1	Redémarrage de l'agent	Cette propriété n'est valide que si la valeur de ContainerManagedEvents est JMS.
PollStartTime	HH = 0 à 23 MM = 0 à 59	HH:MM	Redémarrage du composant	
RepositoryDirectory	<REMOTE> si le courtier est ICS ; sinon tout répertoire local valide.	Pour ICS, la valeur est définie sur <REMOTE> Pour WMQI et WAS, la valeur est *<ProductDir* \repository	Redémarrage de l'agent	
RequestQueue	Nom de file d'attente JMS valide	*<CONNECTORNAME>*/ REQUESTQUEUE	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS
ResponseQueue	Nom de file d'attente JMS valide	*<CONNECTORNAME>* /RESPONSEQUEUE	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
RestartRetryCount	0 à 99	7	Dynamique si ICS ; sinon Redémarrage du composant	
RestartRetryInterval	Une valeur en minutes de 1 à 2147483647	1	Dynamique si ICS ; sinon Redémarrage du composant	

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
ResultsSetEnabled	true ou false	false	Redémarrage du composant	Utilisé uniquement par les connecteurs qui prennent en charge DB2II. Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS, et la valeur de BrokerType est WMQI.
ResultsSetSize	Entier positif	0 (indique que la taille de l'ensemble de résultats est illimitée)	Redémarrage du composant	Utilisé uniquement par les connecteurs qui prennent en charge DB2II. Cette propriété n'est valide que si la valeur de ResultsSetEnabled est true.
RHF2MessageDomain	mrm ou xml	mrm	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS et la valeur de WireFormat est CwXML.
SourceQueue	Tout nom de file d'attente WebSphere MQ	<*CONNECTORNAME*> /SOURCEQUEUE	Redémarrage de l'agent	Cette propriété n'est valide que si la valeur de ContainerManagedEvents est JMS.
SynchronousRequest Queue	Tout nom de file d'attente valide.	<*CONNECTORNAME*> /SYNCHRONOUSREQUEST QUEUE	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
SynchronousRequest Timeout	0 à n'importe quelle valeur (millisecondes)	0	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
SynchronousResponse Queue	Tout nom de file d'attente valide	<*CONNECTORNAME*> /SYNCHRONOUSRESPONSE QUEUE	Redémarrage du composant	Cette propriété n'est valide que si la valeur de DeliveryTransport est JMS.
TivoliMonitorTransaction Performance	true ou false	false	Redémarrage du composant	
WireFormat	CwXML ou CwBO	CwXML	Redémarrage de l'agent	La valeur de cette propriété doit être CwXML si la valeur de RepositoryDirectory n'est pas définie sur <REMOTE>. La valeur doit être CwBO si la valeur de RepositoryDirectory est définie sur <REMOTE>.

Tableau 9. Récapitulatif des propriétés de configuration standard (suite)

Nom de propriété	Valeurs possibles	Valeur par défaut	Méthode de mise à jour	Remarques
WsifSynchronousRequest Timeout	0 à n'importe quelle valeur (millisecondes)	0	Redémarrage du composant	Cette propriété n'est valide que si la valeur de BrokerType est WAS.
XMLNameSpaceFormat	short ou long ou no	short	Redémarrage de l'agent	Cette propriété n'est valide que si la valeur de BrokerType est WMQI ou WAS

Propriétés standard

Cette section décrit les propriétés standard de configuration du connecteur.

AdapterHelpName

La propriété AdapterHelpName est le nom d'un répertoire contenant des fichiers d'aide étendue spécifiques au connecteur. Le répertoire doit figurer dans <*ProductDir*>\bin\Data\App\Help et contenir au moins le répertoire de langue enu_usa. Il peut contenir d'autres répertoires selon les paramètres régionaux.

La valeur par défaut est le nom du modèle s'il est valide, ou elle est vide.

AdminInQueue

La propriété AdminInQueue précise la file d'attente utilisée par le courtier d'intégration pour envoyer des messages administratifs au connecteur.

La valeur par défaut est *<CONNECTORNAME>*/ADMININQUEUE

AdminOutQueue

La propriété AdminOutQueue précise la file d'attente utilisée par le connecteur pour envoyer des messages administratifs au courtier d'intégration.

La valeur par défaut est *<CONNECTORNAME>*/ADMINOUTQUEUE

AgentConnections

La propriété AgentConnections contrôle le nombre de connexions ORB (Object Request Broker) ouvertes à l'initialisation de ORB.

Elle n'est valide que si la valeur de RepositoryDirectory est définie sur <REMOTE> et si la valeur de la propriété DeliveryTransport est MQ ou IDL.

La valeur par défaut de cette propriété est 1.

AgentTraceLevel

La propriété AgentTraceLevel définit le niveau des messages de trace pour le composant spécifique à l'application. Le connecteur fournit tous les messages de trace applicables au niveau de trace défini et à un niveau inférieur.

La valeur par défaut est 0.

ApplicationName

La propriété ApplicationName identifie de façon unique le nom de l'application du connecteur. Ce nom permet à l'administrateur système de surveiller l'environnement d'intégration. Vous devez attribuer une valeur à cette propriété avant d'exécuter le connecteur.

La valeur par défaut est le nom du connecteur.

BiDi.Application

La propriété BiDi.Application précise le format bidirectionnel des données provenant d'une application externe et entrant dans l'adaptateur, sous la forme d'un objet métier pris en charge par cet adaptateur. La propriété définit les attributs bidirectionnels des données de l'application. Ces attributs sont les suivants :

- Type de texte : implicite ou visuel (`I` ou `V`)
- Direction du texte : de gauche à droite ou de droite à gauche (`L` ou `R`)
- Permutation symétrique : activée ou désactivée (`Y` ou `N`)
- Mise en forme (arabe): activée ou désactivée (`S` ou `N`)
- Mise en forme numérique (arabe) : hindi, contextuel, ou nominal (`H`, `C` ou `N`)

Cette propriété n'est
valide que si la valeur de la propriété BiDi.Transformation est définie sur true.

La valeur par défaut est `ILYNN` (implicite, gauche à droite, activé, désactivé, nominal).

BiDi.Broker

La propriété BiDi.Broker précise le format de script bidirectionnel pour les données envoyées depuis l'adaptateur au courtier d'intégration sous la forme d'un objet métier pris en charge. Elle définit les attributs bidirectionnels des données, indiqués sous `BiDi.Application` ci-dessous.

Cette propriété n'est
valide que si la valeur de la propriété BiDi.Transformation est définie sur true. Si la propriété BrokerType est ICS, sa valeur est en lecture seule.

La valeur par défaut est `ILYNN` (implicite, gauche à droite, activé, désactivé, nominal).

BiDi.Metadata

La propriété BiDi.Metadata définit le format bidirectionnel ou les attributs des métadonnées qui sont utilisées par le connecteur pour établir et maintenir un lien vers l'application externe. Les paramètres de l'attribut sont spécifiques à chaque adaptateur qui utiliser des capacités bidirectionnelles. Si votre adaptateur prend en charge le traitement bidirectionnel, voir la section relative aux propriétés spécifiques à l'adaptateur pour plus d'informations.

Cette propriété n'est
valide que si la valeur de la propriété BiDi.Transformation est définie sur true.

La valeur par défaut est `ILYNN` (implicite, gauche à droite, activé, désactivé, nominal).

BiDi.Transformation

La propriété BiDi.Transformation détermine si le système procède ou non à une transformation bidirectionnelle lors de l'exécution.

Si la valeur de la propriété est définie sur `true`, les propriétés BiDi.Application, BiDi.Broker et BiDi.Metadata sont disponibles. Si la valeur de la propriété est définie sur `false`, elles sont cachées.

La valeur par défaut est `false`.

BOTrace

La propriété BOTrace indique si les messages de trace d'objet métier sont activés ou non lors de l'exécution.

Remarque : Ceci ne s'applique que si la propriété AgentTraceLevel est inférieure à 5.

Lorsque le niveau de trace est inférieur à 5, vous pouvez utiliser ces paramètres de ligne de commande pour réinitialiser la valeur de BOTrace.

- Entrez `-xBOTrace=Full` pour afficher tous les attributs d'objets métier.
- Entrez `-xBOTrace=Keys` pour n'afficher que les clés d'objets métier.
- Entrez `-xBOTrace=None` pour désactiver l'affichage des attributs d'objets métier.

La valeur par défaut est `false`.

BrokerType

La propriété BrokerType identifie le type de courtier d'intégration que vous utilisez. Les valeurs possibles sont `ICS`, `WMQI (pour WMQI, WMQIB ou WBIMB)` ou `WAS`.

CharacterEncoding

La propriété CharacterEncoding indique le jeu de codes de caractères utilisé pour mettre en correspondance un caractère (une lettre de l'alphabet, un chiffre ou un signe de ponctuation) et une valeur numérique.

Remarque : Les connecteurs Java n'utilisent pas cette propriété. Les connecteurs C++ utilisent la valeur `ascii7` pour cette propriété.

Par défaut, n'est affiché qu'un sous-ensemble des codages de caractères pris en charge. Pour ajouter d'autres valeurs prises en charge à la liste, vous devez modifier manuellement le fichier \Data\Std\stdConnProps.xml dans le répertoire produit (<*ProductDir*>). Pour plus d'informations, voir l'annexe Connector Configurator de ce guide.

CommonEventInfrastructure

L'infrastructure Common Event Infrastructure (CEI) est une fonction simple de gestion des événements chargée de traiter les événements générés. La propriété CommonEventInfrastructure indique si le CEI doit être appelé lors de l'exécution.

La valeur par défaut est `false`.

CommonEventInfrastructureContextURL

CommonEventInfrastructureContextURL est utilisé pour accéder au serveur WAS qui exécute l'application du serveur CEI (Common Event Infrastructure). Cette propriété précise l'URL à utiliser.

Cette propriété n'est
valide que si la valeur de CommonEventInfrastructure est définie sur `true`.

La valeur par défaut est une zone vide.

ConcurrentEventTriggeredFlows

La propriété ConcurrentEventTriggeredFlows détermine le nombre d'objets métier pouvant être traités simultanément par le connecteur pour la transmission des événements. Vous définissez la valeur de cet attribut sur le nombre d'objets métiers qui sont simultanément mappés et livrés. Par exemple, si vous définissez la valeur de cette propriété sur `5`, cinq objets métier sont traités simultanément.

La définition de cette propriété sur une valeur supérieure à 1 permet au connecteur d'une application source de mapper plusieurs objets métier d'événement en même temps et de les transmettre simultanément à plusieurs instances de collaboration. Cela augmente la rapidité de transmission des objets métier au courtier d'intégration, en particulier si les objets métier utilisent des mappes complexes. L'augmentation du taux d'arrivée des objets métier aux instances de collaboration peut améliorer les performances générales du système.

Pour implémenter le traitement simultané d'un flux entier (d'une application source vers une application cible), vous devez configurer les propriétés suivantes :

- La collaboration doit être configurée de façon à utiliser plusieurs unités d'exécution simultanées, en indiquant pour la propriété `Maximum number of concurrent events` une valeur suffisamment élevée.
- Le composant spécifique à l'application de destination doit être configuré pour traiter les requêtes simultanément. C'est à dire qu'il doit avoir plusieurs unités d'exécution ou être capable d'utiliser le parallélisme de l'agent du connecteur et être configuré pour plusieurs processus. Attribuez une valeur supérieure à 1 à la propriété de configuration Parallel Process Degree.

La propriété ConcurrentEventTriggeredFlows property n'a aucun effet sur l'interrogation du connecteur, laquelle n'a qu'une seule unité d'exécution et est exécutée en série.

La propriété n'est valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE>.

La valeur par défaut est `1`.

ContainerManagedEvents

La propriété ContainerManagedEvents permet à un connecteur activé par JMS et utilisant un magasin d'événements JMS d'effectuer une transmission garantie d'événement, dans laquelle un événement est retiré de la file d'attente source et placé sur la file d'attente cible en tant qu'une transaction JMS.

Lorsque cette propriété est définie sur `JMS`, les propriétés suivantes doivent également être définies pour activer la transmission garantie d'événement :

- PollQuantity = `1` à `500`

- SourceQueue = /SOURCEQUEUE

Vous devez aussi configurer un gestionnaire de données avec les propriétés MimeType et DHClass (classe de gestionnaire de données). Vous pouvez également ajouter DataHandlerConfigMOName (le nom de méta-objet facultatif). Pour définir ces valeurs, utilisez l'onglet **Data Handler** dans Connector Configurator.

Bien que ces propriétés soient spécifiques à l'adaptateur, voici quelques exemples de valeurs :

- MimeType = text\xml
- DHClass = com.crossworlds.DataHandlers.text.xml
- DataHandlerConfigMOName = MO_DataHandler_Default

Les zones qui correspondent à ces valeurs dans l'onglet **Data Handler** ne s'affichent que si vous avez défini la propriété ContainerManagedEvents sur la valeur JMS.

Remarque : Lorsque ContainerManagedEvents a la valeur JMS, le connecteur n'appelle pas sa méthode pollForEvents(), ce qui en désactive la fonctionnalité.

La propriété ContainerManagedEvents n'est valide que si la valeur de la propriété DeliveryTransport est définie sur JMS.

Il n'y a pas de valeur par défaut.

ControllerEventSequencing

La propriété ControllerEventSequencing autorise le séquençage des événements dans le contrôleur du connecteur.

Cette propriété n'est
valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE> (BrokerType égal à ICS).

La valeur par défaut est true.

ControllerStoreAndForwardMode

La propriété ControllerStoreAndForwardMode définit le comportement du contrôleur du connecteur après avoir détecté l'indisponibilité du composant spécifique à l'application cible.

Si cette propriété a la valeur true et que le composant spécifique à l'application cible n'est pas disponible lorsqu'un événement atteint l'ICS, le contrôleur du connecteur empêche la requête d'accéder au composant spécifique à l'application. Lorsque le composant spécifique à l'application redevient opérationnel, le contrôleur lui envoie la requête.

Toutefois, si le composant d'application cible devient indisponible après que le contrôleur du connecteur lui a envoyé la requête d'appel de service, celle-ci échoue.

Si cette propriété a la valeur false, le contrôleur du connecteur met toutes les requêtes d'appels de service en échec dès qu'il détecte l'indisponibilité du composant spécifique à l'application.

Cette propriété n'est
valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE> (la valeur de la propriété BrokerType est ICS).

La valeur par défaut est `true`.

ControllerTraceLevel

La propriété ControllerTraceLevel définit le niveau des messages de trace pour le contrôleur de connecteur.

La propriété n'est valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE>.

La valeur par défaut est `0`.

DeliveryQueue

La propriété DeliveryQueue définit la file d'attente utilisée par le connecteur pour envoyer des objets métier au courtier d'intégration.

Cette propriété n'est
valide que si la valeur de la propriété DeliveryTransport est définie sur `JMS`.

La valeur par défaut est *<CONNECTORNAME>*`/DELIVERYQUEUE`.

DeliveryTransport

La propriété DeliveryTransport spécifie le mécanisme de transfert pour la transmission des événements. Les valeurs possibles sont `MQ` pour WebSphere MQ, `IDL` pour CORBA IIOP et `JMS` pour Java Messaging Service.

- Si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE>, la valeur de la propriété DeliveryTransport peut être `MQ`, `IDL` ou `JMS`, et la valeur par défaut est `IDL`.
- Si la valeur de la propriété RepositoryDirectory est un répertoire local, l'unique valeur possible est `JMS`.

Si la valeur de la propriété RepositoryDirectory est `MQ` ou `IDL`, le connecteur envoie des requêtes d'appel de service et des messages administratifs par CORBA IIOP.

Si la valeur de la propriété DeliveryTransport est `MQ`, vous pouvez définir le paramètre de ligne de commande `WhenServerAbsent` dans le script de démarrage de l'adaptateur, de façon à indiquer si l'adaptateur doit se mettre en pause ou se fermer lorsque InterChange Server est arrêté.

- Entrez `WhenServerAbsent=pause` pour mettre l'adaptateur en pause lorsque ICS n'est pas disponible.
- Entrez `WhenServerAbsent=shutdown` pour arrêter l'adaptateur lorsque ICS n'est pas disponible.

WebSphere MQ et IDL

Utilisez WebSphere MQ plutôt que IDL pour le transfert d'événement, sauf si vous ne devez avoir qu'un seul produit. WebSphere MQ présente les avantages suivants par rapport à IDL :

- Communication asynchrone :
  WebSphere MQ permet au composant spécifique à l'application d'interroger et de stocker de manière permanente les événements, même lorsque le serveur n'est pas disponible.
- Performance côté serveur :
  WebSphere MQ offre plus de rapidité du côté du serveur. En mode optimisé, WebSphere MQ ne stocke que le pointeur désignant un événement dans la base de données du référentiel, tandis que l'événement correspondant reste dans la file d'attente de WebSphere MQ. Ceci empêche d'écrire des événements potentiellement volumineux dans la base de données du référentiel.
- Performance côté agent :
  WebSphere MQ offre plus de rapidité du côté du composant spécifique à l'application. Avec WebSphere MQ, l'unité d'exécution d'interrogation du connecteur sélectionne un événement, le place dans la file d'attente du connecteur, puis sélectionne l'événement suivant. Cette méthode est plus rapide que celle d'IDL, dans laquelle l'unité d'exécution d'interrogation du connecteur doit sélectionner un événement, accéder au réseau dans le processus du serveur, stocker l'événement de manière permanente dans la base de données du référentiel, puis sélectionner l'événement suivant.

JMS

Le mécanisme de transfert JMS active la communication entre le connecteur et l'architecture du connecteur client à l'aide de Java Messaging Service (JMS).

Si vous sélectionnez JMS en tant que transfert, d'autres propriétés JMS supplémentaires telles que `jms.MessageBrokerName`, `jms.FactoryClassName`, `jms.Password` et `jms.UserName` apparaissent dans Connector Configurator. Les propriétés `jms.MessageBrokerName` et `jms.FactoryClassName` sont obligatoires pour ce transfert.

Il peut y avoir une limitation de mémoire si vous utilisez le mécanisme de transfert JMS pour un connecteur dans l'environnement suivant :

- AIX 5.0
- WebSphere MQ 5.3.0.1
- ICS est le courtier d'intégration

Dans cet environnement, vous rencontrerez peut être des difficultés pour démarrer simultanément le contrôleur du connecteur (du côté du serveur) et le connecteur (du côté du client), en raison de l'utilisation de la mémoire dans le client WebSphere MQ. Si votre installation utilise une taille de segment de processus inférieure à 768 Mo, définissez la variable et la propriété suivantes :

- Définissez la variable d'environnement LDR_CNTRL dans le script CWSharedEnv.sh.

  Ce script se trouve dans le répertoire \bin sous le répertoire produit (<*ProductDir*>). A l'aide d'un éditeur de texte, ajoutez la ligne suivante à la première ligne du script CWSharedEnv.sh :

  ```
  export LDR_CNTRL=MAXDATA=0x30000000
  ```

Cette ligne de commande restreint l'utilisation du segment de mémoire à un maximum de 768 Mo (3 segments * 256 Mo). Si la mémoire du processus dépasse cette limite, un échange de pages peut se produire, ce qui peut affecter les performances de votre système.

- Définissez la valeur de la propriété IPCCBaseAddress sur 11 ou 12. Pour plus d'informations sur cette propriété, voir le document *System Installation Guide for UNIX*.

DuplicateEventElimination

Lorsque la valeur de cette propriété est `true`, un connecteur activé pour JMS peut vérifier que des doublons ne sont pas transmis à la file d'attente de transmission. Pour utiliser cette fonction, le connecteur doit recevoir pendant son développement un identificateur d'événement unique défini en tant qu'attribut ObjectEventId de l'objet métier dans le code spécifique à l'application.

Remarque : Lorsque la valeur de cette propriété est `true`, la propriété MonitorQueue doit être activée pour garantir la livraison de l'événement.

La valeur par défaut est `false`.

EnableOidForFlowMonitoring

Lorsque la valeur de cette propriété est `true`, l'exécution de l'adaptateur marque le ObjectEventID entrant en tant que clé étrangère pour la surveillance du flot.

La propriété n'est valide que si la propriété BrokerType est définie sur ICS.

La valeur par défaut est `false`.

FaultQueue

Si le connecteur rencontre une erreur lors du traitement d'un message, il transmet ce message à la file d'attente indiquée dans la propriété FaultQueue (accompagné d'un indicateur de statut et d'une description de l'incident).

La valeur par défaut est *<CONNECTORNAME>*`/FAULTQUEUE`.

jms.FactoryClassName

La propriété jms.FactoryClassName indique le nom de classe à instancier pour un fournisseur JMS. Cette propriété doit être définie si la valeur de la propriété DeliveryTransport property est `JMS`.

La valeur par défaut est `CxCommon.Messaging.jms.IBMMQSeriesFactory`.

jms.ListenerConcurrency

La propriété jms.ListenerConcurrency indique le nombre de programmes d'écoute simultanés pour le contrôleur JMS. Elle précise le nombre d'unités d'exécution qui extraient et traitent les messages simultanément, dans un contrôleur.

Cette propriété n'est
valide que si la valeur de la propriété jms.OptimizedTransport est `true`.

La valeur par défaut est 1.

jms.MessageBrokerName

jms.MessageBrokerName précise le nom de courtier à utiliser pour le fournisseur JMS. Vous devez définir cette propriété de connecteur si vous précisez JMS en tant que mécanisme de transfert (dans la propriété DeliveryTransport).

Lorsque vous vous connectez à un courtier de messages éloigné, cette propriété exige les valeurs suivantes :
QueueMgrName:*Channel*:*HostName*:*PortNumber*
où :
QueueMgrName est le nom du gestionnaire de files d'attente.
Channel est le canal utilisé par le client.
HostName est le nom de la machine sur laquelle doit résider le gestionnaire de files d'attente.
PortNumber est le numéro de port sur lequel écoute le gestionnaire de files d'attente.

Par exemple :
```
jms.MessageBrokerName = WBIMB.Queue.Manager:CHANNEL1:RemoteMachine:1456
```

La valeur par défaut est crossworlds.queue.manager. Utilisez la valeur par défaut lorsque vous vous connectez à un courtier de messages local.

jms.NumConcurrentRequests

La propriété jms.NumConcurrentRequests indique le nombre maximal de requêtes d'appel de service pouvant être envoyées simultanément à un connecteur. Lorsque ce nombre maximal est atteint, les nouveaux appels de service sont bloqués et mis en attente de traitement.

La valeur par défaut est 10.

jms.Password

La propriété jms.Password indique le mot de passe défini pour le fournisseur JMS. Cette valeur est facultative.

Il n'y a pas de valeur par défaut.

jms.TransportOptimized

La propriété jms.TransportOptimized détermine si l'opération en cours (WIP) est optimisée. Pour l'optimiser, vous devez disposer d'un fournisseur WebSphere MQ. Pour que le WIP optimisé fonctionne, le fournisseur de messagerie doit pouvoir :

1. Lire un message sans le retirer de la file d'attente
2. Supprimer un message ayant un ID donné sans transférer le message entier dans l'espace mémoire du réceptionnaire
3. Lire un message en utilisant un ID donné (nécessaire pour la récupération)
4. Déterminer à quel moment apparaissent les événements qui n'ont pas été lus.

Les API JMS ne peuvent par être utilisées pour le WIP optimisé car elles ne remplissent pas les conditions 2 et 4 ci-dessus. Les API MQ Java remplissent les quatre conditions et sont donc requises pour le WIP optimisé.

Cette propriété n'est
valide que si la valeur de DeliveryTransport est JMS et la valeur de BrokerType est ICS.

La valeur par défaut est `false`.

jms.UserName

La propriété jms.UserName indique le nom d'utilisateur du fournisseur JMS. Cette valeur est facultative.

Il n'y a pas de valeur par défaut.

JvmMaxHeapSize

La propriété JvmMaxHeapSize indique la taille de segment maximale pour l'agent (en megaoctets).

La propriété n'est valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE>.

La valeur par défaut est 128 Mo.

JvmMaxNativeStackSize

La propriété JvmMaxNativeStackSize indique l'espace mémoire natif maximal pour l'agent (en kilo-octets).

La propriété n'est valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE>.

La valeur par défaut est 128 Ko.

JvmMinHeapSize

La propriété JvmMinHeapSize indique la taille de segment minimale pour l'agent (en megaoctets).

La propriété n'est valide que si la valeur de la propriété RepositoryDirectory est définie sur <REMOTE>.

La valeur par défaut est 1 Mo.

ListenerConcurrency

La propriété ListenerConcurrency prend en charge le traitement de plusieurs unités d'exécution dans WebSphere MQ Listener lorsque ICS est le courtier d'intégration. Elle permet l'écriture par lots de plusieurs événements sur la base de données, ce qui améliore les performances du système.

Cette propriété n'est
valide que pour les connecteurs qui utilisent le transfert MQ. La valeur de la propriété DeliveryTransport doit être définie sur `MQ`.

La valeur par défaut est 1.

Locale

La propriété Locale indique le code de langue, le pays ou le territoire et, le cas échéant, le jeu de codes de caractères associé. La valeur de cette propriété détermine les conventions culturelles telles que le classement et l'ordre de tri des données, les formats de date et d'heure, ainsi que les symboles monétaires utilisés.

Le format d'un nom d'environnement local est le suivant :
ll_TT.codeset

où :
ll est un code de langue à deux caractères (en minuscules)
TT est un code pays ou territoire à deux caractères (en majuscules)
codeset est le nom du jeu de codes de caractères associé (facultatif).

Par défaut, n'est affiché qu'un sous-ensemble des paramètres régionaux pris en charge. Pour ajouter d'autres valeurs prises en charge à la liste, modifiez le fichier \Data\Std\stdConnProps.xml dans le répertoire <*ProductDir*>\bin. Pour plus d'informations, voir l'annexe Connector Configurator de ce guide.

Si le connecteur n'a pas été internationalisé, la seule valeur correcte pour cette propriété est `en_US`. Pour déterminer si un connecteur spécifique a été internationalisé, consultez le guide utilisateur de l'adaptateur.

La valeur par défaut est `en_US`.

LogAtInterchangeEnd

La propriété LogAtInterchangeEnd indique s'il faut consigner les erreurs dans le journal du courtier d'intégration.

La consignation des erreurs dans le journal active également la notification par courrier électronique qui, lorsque des erreurs ou erreurs fatales ont lieu, génère des messages électroniques pour le destinataire spécifié par la valeur MESSAGE_RECIPIENT dans le fichier InterchangeSystem.cfg. Par exemple, lorsque la connexion entre un connecteur et son application est interrompue, si la valeur de LogAtInterChangeEnd est `true`, un courrier électronique est envoyé au destinataire indiqué.

Cette propriété n'est
valide que si la valeur de la propriété RespositoryDirectory est définie sur <REMOTE> (la valeur de BrokerType est ICS).

La valeur par défaut est `false`.

MaxEventCapacity

La propriété MaxEventCapacity indique le nombre maximal d'événements contenus dans la mémoire tampon du contrôleur. Cette propriété est utilisée par la fonctionnalité de contrôle de flux.

Cette propriété n'est
valide que si la valeur de la propriété RespositoryDirectory est définie sur <REMOTE> (la valeur de BrokerType est ICS).

La valeur peut être un nombre entier positif compris entre 1 et 2147483647.

La valeur par défaut est 2147483647.

MessageFileName

La propriété MessageFileName indique le nom du fichier de messages du connecteur. L'emplacement standard de ce fichier de messages est

\connectors\messages, dans le répertoire produit. Si le fichier de messages n'est pas situé à l'emplacement standard, indiquez son nom dans un chemin d'accès absolu.

S'il n'existe pas de fichier de messages, le connecteur utilise `InterchangeSystem.txt` comme fichier de messages. Ce fichier est situé dans le répertoire produit.

Remarque : Pour déterminer si un connecteur a son propre fichier de messages, reportez-vous au guide d'utilisation de l'adaptateur.

La valeur par défaut est `InterchangeSystem.txt`.

MonitorQueue

La propriété MonitorQueue indique la file d'attente logique utilisée par le connecteur pour surveiller les événements en double.

Elle n'est valide que si la valeur de la propriété DeliveryTransport est `JMS` et la valeur de DuplicateEventElimination est `true`.

La valeur par défaut est `<CONNECTORNAME>/MONITORQUEUE`

OADAutoRestartAgent

La propriété OADAutoRestartAgent indique si le connecteur utilise la fonction de redémarrage automatique et éloigné. Cette fonction utilise le démon d'activation d'objets (OAD, Object Activation Daemon) déclenché par WebSphere MQ pour redémarrer le connecteur après un arrêt anormal ou pour démarrer un connecteur éloigné à partir du moniteur système.

Cette propriété doit avoir la valeur `true` pour que la fonction de redémarrage automatique et à distance soit activée. Pour plus d'informations sur la configuration de la fonction de l'OAD déclenché par WebSphere MQ, voir le document *Installation Guide for Windows* ou *for UNIX*.

Cette propriété n'est
valide que si la valeur de la propriété RespositoryDirectory est définie sur <REMOTE> (la valeur de BrokerType est ICS).

La valeur par défaut est `false`.

OADMaxNumRetry

La propriété OADMaxNumRetry indique le nombre maximal de tentatives de redémarrage du connecteur après un arrêt anormal, automatiquement tentées par l'OAD déclenché par WebSphere MQ. Pour cela, la propriété OADAutoRestartAgent doit avoir la valeur `true`.

Cette propriété n'est
valide que si la valeur de la propriété RespositoryDirectory est définie sur <REMOTE> (la valeur de BrokerType est ICS).

La valeur par défaut est `1000`.

OADRetryTimeInterval

La propriété OADRetryTimeInterval indique pour l'OAD déclenché par WebSphere MQ la durée en minutes entre les tentatives de relance. Si l'agent du connecteur ne redémarre pas durant cet intervalle, le contrôleur du connecteur demande à l'OAD de redémarrer l'agent du connecteur. L'OAD répète cette opération autant de fois que spécifié par la propriété OADMaxNumRetry. Pour cela, la propriété OADAutoRestartAgent doit avoir la valeur `true`.

Cette propriété n'est
valide que si la valeur de la propriété RespositoryDirectory est définie sur <REMOTE> (la valeur de BrokerType est ICS).

La valeur par défaut est `10`.

PollEndTime

La propriété PollEndTime indique l'heure d'arrêt de l'interrogation de la file d'attente des événements. Le format est *HH:MM*, dans lequel *HH* représente les heures (de 0 à 23) et *MM* représente les secondes (de 0 à 59).

Vous devez saisir une valeur correcte pour cette propriété. La valeur par défaut est `HH:MM` sans valeur indiquée, mais elle doit être précisée.

Si l'exécution de l'adaptateur détecte :

- que PollStartTime est défini et PollEndTime n'est pas défini, ou
- que PollEndTime est défini et PollStartTime n'est pas défini,

l'interrogation utilisera la valeur configurée pour la propriété PollFrequency.

PollFrequency

La propriété PollFrequency indique la durée (en millisecondes) entre la fin de la dernière interrogation et le début de la suivante. Il ne s'agit pas de l'intervalle entre chaque interrogation. Le principe est le suivant :

- Lancez une interrogation pour obtenir le nombre d'objets spécifié par la propriété PollQuantity.
- Traitez ces objets. Pour certains connecteurs, ceci peut se faire en partie sur des unités d'exécution séparées, qui s'exécutent de manière asynchrone jusqu'à l'interrogation suivante.
- Attendez pendant l'intervalle indiqué par la propriété PollFrequency.
- Répétez le cycle.

La valeurs suivantes sont valides pour cette propriété :

- Un nombre de millisecondes (un entier positif).
- Le mot `no`, pour que le connecteur n'émette pas d'interrogation. Saisissez le mot en minuscules.
- Le mot `key`, pour que le connecteur émette des interrogations uniquement lorsque vous tapez la lettre `p` dans la fenêtre d'invite de commande du connecteur. Saisissez le mot en minuscules.

La valeur par défaut est `10000`.

Important : Certains connecteurs sont limités dans l'utilisation de cette propriété. Ces restrictions sont décrites dans le chapitre sur l'installation et la configuration de l'adaptateur.

PollQuantity

La propriété PollQuantity désigne le nombre d'éléments de l'application pour lesquels le connecteur doit émettre des interrogations. Si l'adaptateur dispose d'une propriété spécifique au connecteur pour définir le nombre d'interrogations, la valeur définie dans cette propriété spécifique remplace la valeur de la propriété standard.

Cette propriété n'est
valide que si la valeur de la propriété `DeliveryTransport` est `JMS`, et si la propriété `ContainerManagedEvents` a une valeur.

Un message électronique est également considéré comme étant un événement. Lorsqu'il est interrogé pour un courrier électronique, le connecteur agit comme suit :

- Lorsqu'il est interrogé une fois, le connecteur détecte le corps du message, qu'il lit comme une pièce jointe. Comme aucun gestionnaire de données n'a été spécifié pour ce type mime, il ignorera le message.
- Le connecteur traite la première pièce jointe BO. Le gestionnaire de données est disponible pour ce type MIME : l'objet métier est envoyé à Visual Test Connector.
- Lorsqu'il est interrogé pour la deuxième fois, le connecteur traite la deuxième pièce jointe BO. Le gestionnaire de données est disponible pour ce type MIME : l'objet métier est envoyé à Visual Test Connector.
- Une fois acceptée, la troisième pièce jointe BO peut être transmise.

PollStartTime

La propriété PollStartTime indique l'heure de démarrage de l'interrogation de la file d'attente des événements. Le format est *HH:MM*, dans lequel *HH* représente les heures (de 0 à 23) et *MM* représente les secondes (de 0 à 59).

Vous devez saisir une valeur correcte pour cette propriété. La valeur par défaut est `HH:MM` sans valeur indiquée, mais elle doit être modifiée.

Si l'exécution de l'adaptateur détecte :

- que PollStartTime est défini et PollEndTime n'est pas défini, ou
- que PollEndTime est défini et PollStartTime n'est pas défini,

l'interrogation utilisera la valeur configurée pour la propriété PollFrequency.

RepositoryDirectory

La propriété RepositoryDirectory indique l'emplacement du référentiel à partir duquel le connecteur lit les schémas XML qui stockent les métadonnées pour la définition des objets métier.

Si ICS est le courtier d'intégration, cette valeur doit être définie sur <REMOTE>, car le connecteur obtient ces informations à partir du référentiel d'InterChange Server.

Lorsque le courtier d'intégration est un courtier de message WebSphere ou WAS, cette valeur est définie par défaut sur <*ProductDir*>\repository. Toutefois, elle peut être définie sur tout nom valide de répertoire.

RequestQueue

La propriété RequestQueue précise la file d'attente utilisée par le courtier d'intégration pour envoyer des objets métier au connecteur.

Cette propriété n'est
valide que si la valeur de la propriété DeliveryTransport est JMS.

La valeur par défaut est <*CONNECTORNAME*>/REQUESTQUEUE.

ResponseQueue

La propriété ResponseQueue désigne la file d'attente de réponses JMS, qui transmet un message de réponse depuis l'architecture du connecteur vers le courtier d'intégration. Lorsqu'ICS est le courtier d'intégration, le serveur envoie la requête et attend un message de réponse dans la file d'attente de réponses JMS.

Cette propriété n'est
valide que si la valeur de la propriété DeliveryTransport est définie sur JMS.

La valeur par défaut est <*CONNECTORNAME*>/RESPONSEQUEUE.

RestartRetryCount

La propriété RestartRetryCount indique le nombre de tentatives de redémarrage du connecteur. Lorsqu'elle est utilisée pour un connecteur parallèle, cette propriété indique le nombre de tentatives de redémarrage du composant spécifique à l'application du connecteur client par le composant spécifique à l'application du connecteur maître.

La valeur par défaut est 7.

RestartRetryInterval

La propriété RestartRetryInterval indique l'intervalle en minutes pendant lequel le connecteur tente de se redémarrer. Lorsqu'elle est utilisée pour un connecteur parallèle, cette propriété indique l'intervalle entre les tentatives de redémarrage du composant spécifique à l'application du connecteur client par le composant spécifique à l'application du connecteur maître.

Les valeurs possibles vont de 1 à 2147483647.

La valeur par défaut est 1.

ResultsSetEnabled

La propriété ResultsSetEnabled active ou désactive la prise en charge de l'ensemble des résultats lorsque Information Integrator est actif. Cette propriété ne peut être utilisée que si l'adaptateur prend en charge DB2 Information Integrator.

Cette propriété n'est
valide que si la valeur de DeliveryTransport est JMS et la valeur de BrokerType est WMQI.

La valeur par défaut est false.

ResultsSetSize

La propriété ResultsSetSize définit le nombre maximum d'objets métier pouvant être retournés à Information Integrator. Cette propriété ne peut être utilisée que si l'adaptateur prend en charge DB2 Information Integrator.

Cette propriété n'est
valide que si la propriété ResultsSetEnabled est définie sur `true`.

La valeur par défaut est `0`. Ce qui signifie que la taille de l'ensemble de résultats est illimitée.

RHF2MessageDomain

La propriété RHF2MessageDomain vous permet de configurer la valeur du nom de domaine de la zone dans l'en-tête JMS. Lorsque les données sont envoyées à un courtier de message WebSphere par transfert JMS, l'architecture de l'adaptateur écrit les informations de l'en-tête JMS, avec un nom de domaine et une valeur fixe `mrm`. Un nom de domaine configurable vous permet d'analyser comment le courtier de messages WebSphere traite les données de message.

Voici un exemple d'en-tête :

```
<mcd><Msd>mrm</Msd><Set>3</Set><Type>
Retek_POPhyDesc</Type><Fmt>CwXML</Fmt></mcd>
```

Cette propriété n'est
valide que si la valeur de BrokerType est WMQI ou WAS. De plus, elle n'est valide que si la valeur de la propriété DeliveryTransport est `JMS` et si la valeur de WireFormat est `CwXML`.

Les valeurs possibles sont `mrm` et `xml`. La valeur par défaut est `mrm`.

SourceQueue

La propriété SourceQueue désigne la file d'attente source JMS de l'architecture du connecteur qui assure la transmission garantie d'événements pour les connecteur activés par JMS qui utilisent un magasin d'événements JMS. Pour plus d'informations, voir «ContainerManagedEvents», à la page 65.

Cette propriété n'est
valide que si la valeur de DeliveryTransport est `JMS` et si une valeur est indiquée pour ContainerManagedEvents.

La valeur par défaut est *<CONNECTORNAME>*`/SOURCEQUEUE`.

SynchronousRequestQueue

La propriété SynchronousRequestQueue transmet les messages de requête qui requièrent une réponse synchrone depuis l'architecture du connecteur vers le courtier. Cette file d'attente n'est nécessaire que si le connecteur utilise l'exécution synchrone. Avec l'exécution synchrone, l'architecture du connecteur envoie un message à la file d'attente de requêtes synchrones et attend une réponse du courtier sur la file d'attente de réponse synchrone. La réponse envoyée au connecteur a un ID de corrélation qui correspond à l'ID du message d'origine.

Cette propriété n'est
valide que si la valeur de la propriété DeliveryTransport est définie sur `JMS`.

La valeur par défaut est *<CONNECTORNAME>*/SYNCHRONOUSREQUESTQUEUE

SynchronousRequestTimeout

La propriété SynchronousRequestTimeout indique la durée d'attente en millisecondes d'une réponse à une requête synchrone. Si la réponse n'est pas reçue dans l'intervalle de temps indiqué, le connecteur transfère le message de requête synchrone original, accompagné d'un message d'erreur, dans la file d'attente des erreurs.

Cette propriété n'est
valide que si la valeur de la propriété DeliveryTransport est définie sur JMS.

La valeur par défaut est 0.

SynchronousResponseQueue

La propriété SynchronousResponseQueue transmet les messages de réponse à une requête synchrone entre le courtier et l'architecture du connecteur. Cette file d'attente n'est nécessaire que si le connecteur utilise l'exécution synchrone.

Cette propriété n'est
valide que si la valeur de la propriété DeliveryTransport est définie sur JMS.

La valeur par défaut est *<CONNECTORNAME>*/SYNCHRONOUSRESPONSEQUEUE

TivoliMonitorTransactionPerformance

La propriété TivoliMonitorTransactionPerformance indique si IBM Tivoli Monitoring for Transaction Performance (ITMTP) est appelé lors de l'exécution.

La valeur par défaut est false.

WireFormat

La propriété WireFormat précise le format de message sur le transfert :

- Si la valeur de la propriété RepositoryDirectory est un répertoire local, la valeur est CwXML.
- Si la valeur de la propriété RepositoryDirectory est un répertoire éloigné, la valeur est CwBO.

WsifSynchronousRequestTimeout

La propriété WsifSynchronousRequestTimeout indique la durée d'attente en millisecondes d'une réponse à une requête synchrone. Si la réponse n'est pas reçue dans l'intervalle de temps indiqué, le connecteur transfère le message de requête synchrone original, accompagné d'un message d'erreur, dans la file d'attente des erreurs.

Cette propriété n'est
valide que si la valeur de BrokerType est définie sur WAS.

La valeur par défaut est 0.

XMLNameSpaceFormat

La propriété XMLNameSpaceFormat précise des espaces de nom courts ou longs dans le format XML des définitions d'objet métier.

Cette propriété n'est
valide que si la valeur de BrokerType est WMQI ou WAS.

La valeur par défaut est `short`.

Index

A

adaptateur iSeries
  arrêt du connecteur 27
  compatibilité du courtier 5
  conditions requises 6
  configuration 9
  démarrage 25
  environnement 5
  Généralités 1
  installation 5
  installation et fichiers associés 7
  opérations 4
  platesformes 6
  résolution des incidents 49
après l'installation
  tâches 7
ASI de l'attribut
  indication 45
ASI de l'objet métier
  indication 46

B

Business Object Designer
  exécution 39

C

configuration
  remplissage 24
Connector Configurator
  démarrage 10
  exécution à partir de System Manager 11
  exécution en mode autonome 10
  Généralités 9
  utilisation en environnements globalisés 25
consignation 49

D

Data conversion
  kit d'outils iSeries ou AS/400 38

F

fichier de configuration
  création à partir d'un modèle spécifique au connecteur 14
  création d'un nouveau 14
  définition des propriétés 16
  enregistrement 24
  modification 24
  remplissage 16
  utilisation d'un fichier existant 15
files d'attente de données
  Généralités 3
Fonctionnement de l'adaptateur 3

G

gestion des erreurs 49
gestionnaires de données 23

I

instances du connecteur
  création de plusieurs 27

M

mappes 22
messagerie 23
Messages de traçage 49
méta-objets
  configuration en vue de l'interrogation 34
Métadonnées
  définition 29
modèle de propriété spécifique au connecteur
  création 11

O

Object Discovery Agent (ODA)
  démarrage 39
  Généralités 39
objets métier
  création et modification 39
  génération 44
  génération de définitions 39
  indication d'informations 45
  indication des définitions prises en charge 20
  indication du texte de l'application au niveau de l'attribut 37
  présentation 29
  présentation de la structure 29
  spécification des propriétés d'attribut 36
  structure de la file d'attente de données 34
  structure des programmes RPG 30
  téléchargement 47
  traitement 3

P

propriétés de configuration spécifiques à l'application
  définition 19
propriétés de configuration standard pour les connecteurs 53
propriétés spécifiques au connecteur 18
propriétés standard du connecteur
  configuration 18

R

Ressources 23

S

Structure de fichiers installée 7
systèmes iSeries et AS/400
  Généralités 1

V

valeurs des fichiers journaux/fichiers de trace
  définition 23

Informations légales

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays.

Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM EMEA Director of Licensing
IBM Europe Middle-East Africa
Tour Descartes
La Défense 5
2, avenue Gambetta
92066 - Paris-La Défense CEDEX
France

Pour le Canada, veuillez adresser votre courrier à :

IBM Director of Commercial Relations
IBM Canada Ltd.
3600 Steeles Avenue East
Markham, Ontario
L3R 9Z7
Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues par écrit à l'adresse suivante :

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales.

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT, EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFAÇON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Il est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM conformément aux dispositions de l'ICA, des Conditions internationales d'utilisation des logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins

illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Le présent logiciel contient des exemples de programmes en langage source, destinés à illustrer les techniques de programmation sur différentes platesformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Informations sur les interfaces de programmation

Les informations sur les interfaces de programmation ont pour objectif de vous aider à créer des logiciels d'application à l'aide de ce programme.

Les interfaces de programmation génériques vous permettent de créer des logiciels d'application qui obtiennent les services des outils de ce programme.

Cependant, ces informations peuvent également contenir des informations sur le diagnostic, la modification et le réglage. Ces informations vous permettent d'exécuter le débogage de votre logiciel d'application.

Avertissement : N'utilisez pas ces informations sur le diagnostic, la modification et le réglage comme interface de programmation car elles sont susceptibles de changer.

Marques et marques de service

Les termes qui suivent sont des marques d'International Business Machines Corporation aux Etats-Unis et/ou dans certains autres pays :

i5/OS
IBM
le logo IBM
AIX
AIX 5L
CICS
CrossWorlds
DB2
DB2 Universal Database
Domino
HelpNow
IMS
Informix
iSeries
Lotus

Lotus Notes
MQIntegrator
MQSeries
MVS
Notes
OS/400
Passport Advantage
pSeries
Redbooks
SupportPac
WebSphere
z/OS

Java et toutes les marques incluant Java sont des marques de Sun Microsystems, Inc. aux Etats-Unis et/ou dans certains autres pays.

Microsoft, Windows, Windows NT et le logo Windows sont des marques de Microsoft Corporation aux Etats-Unis et/ou dans certains autres pays.

Intel, le logo Intel, Intel Inside, le logo Intel Inside, Intel Centrino, le logo Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium et Pentium sont des marques ou marques déposées de Intel Corporation ou de ses filiales, aux Etats-Unis et dans d'autres pays.

UNIX est une marque enregistrée de The Open Goup aux Etats-Unis et/ou dans certains autres pays.

Linux est une marque de Linus Torvalds aux Etats-Unis et/ou dans certains autres pays.

Les autres noms de sociétés, de produits et de services peuvent appartenir à des tiers.

Ce produit inclut un logiciel développé par Eclipse Project (http://www.eclipse.org/).

WebSphere Business Integration Adapters, Version 6.0

WebSphere Business Integration Adapter Framework V2.4.0