IBM WebSphere Business Integration Server Express e
Express
Plus

Guida gestore dati

versione 44

Nota!

Prima di utilizzare queste informazioni e il relativo prodotto, leggere attentamente le informazioni riportate in "Informazioni particolari" a pagina 233.

22 Aprile 2005

Questa edizione del documento si applica a IBM WebSphere Business Integration Server Express versione *4.4.0*, IBM WebSphere Business Integration Server Express Plus versione 4.4.0, Toolset Express versione *4.4.0* e a tutte le versioni e le modifiche successive se non altrimenti indicato nelle nuove edizioni.

Per inviare commenti sulla documentazione, utilizzare il seguente indirizzo e-mail: doc-comments@us.ibm.com. Siamo in attesa di ricevere i vostri commenti.

L'IBM può utilizzare o divulgare le informazioni ricevute dagli utenti secondo le modalità ritenute appropriate, senza nessun obbligo nei loro confronti.

Indice

Informazioni

I prodotti IBM(R) WebSphere(R) Business Integration Server Express e IBM(R) WebSphere(R) Business Integration Server Express Plus includono i seguenti componenti: Interchange Server Express, Toolset Express associato, CollaborationFoundation e una serie di adattatori di integrazione software. Gli strumenti di Toolset Express consentono di creare, modificare e gestire i processi di business. E' possibile scegliere tra diversi adattatori preconfigurati per i processi di business che utilizzano le applicazioni. La maschera dei processi standard, CollaborationFoundation, consente di creare rapidamente processi personalizzati.

Questo capitolo descrive i gestori dati distribuiti e le prestazioni dei gestori dati personalizzati.

Se non diversamente indicato, tutte le informazioni riportate in questo manuale sono valide per IBM WebSphere Business Integration Server Express e IBM WebSphere Business Integration Server Express Plus. Il termine "WebSphere Business Integration Server Express" e le relative varianti fa riferimento a entrambi i prodotti.

A chi è rivolto questo manuale

Questo manuale è diretto a consulenti e clienti. E' possibile familiarizzare con gli oggetti di business e gli oggetti metadata. Per l'utilizzo del gestore dati XML, è necessario familiarizzare con i documenti XML, con gli standard XML correnti e con SAX (Simple API for XML). Per l'utilizzo del gestore dati EDI, è necessario familiarizzare con i documenti EDI e con gli standard EDI correnti. Se si ha intenzione di espandere la libreria gestori dati, sarebbe necessario essere esperti nella programmazione in Java.

Documentazione correlata

La serie completa di manuali descrive le funzioni e i componenti comuni a tutte le installazioni di WebSphere Business Integration Server Express e WebSphere Business Integration Server Express Plus e comprende i materiali di riferimento per componenti specifici.

E' possibile scaricare, installare e visualizzare la documentazione sul sito Web al seguente indirizzo: http://www.ibm.com/websphere/wbiserverexpress/infocenter

Nota: Le informazioni rilevanti su questo prodotto sono disponibili nel documento Supporto tecnico note tecniche e informazioni rapide emesso in seguito alla pubblicazione di questo manuale. Tali documenti sono disponibili sul sito Web di supporto di WebSphere Business Integration all'indirizzo http://www.ibm.com/software/integration/websphere/support/. Selezionare l'area dei componenti di interesse e passare alle sezioni note tecniche e informazioni rapide.

Convenzioni tipografiche

Questo manuale utilizza le seguenti convenzioni:

`carattere courier`	Indica un valore letterale, come il nome di un comando, il nome di un file, informazioni immesse dall'utente o le informazioni visualizzate sullo schermo.
nuovo termine	Indica un termine nuovo la prima volta che viene visualizzato.
corsivo, *`corsivo`*	Indica un nome variabile i un riferimento incrociato.
riquadro blu	Uno schema blu, visibile solo quando si visualizza il manuale in linea, indica un collegamento ipertestuale a riferimento incrociato. Fare clic all'interno del riquadro per passare all'oggetto del riferimento.
{ }	Su una riga della sintassi, le parentesi graffe racchiudono una serie di opzioni di cui ne va selezionata solo una.
`[ ]`	In una riga di sintassi, le parentesi quadre racchiudono un parametro facoltativo.
`...`	In una riga di sintassi, le ellissi indicano una ripetizione del parametro precedente. Ad esempio, `option[,...]` significa che è possibile inserire più opzioni separate da virgola.
`< >`	In una convenzione di denominazione, le parentesi angolari racchiudono elementi individuali di un nome per distinguerli tra di loro, come ad esempio in `<nome_server><`*`nome_connettore`*`>tmp.log`.
/, \	In questo documento, le barre rovesciate (\) vengono utilizzare come convenzione per i percorsi di directory. Per le installazioni su sistemi Linux, è necessario sostituire i caratteri barra inversa con le barre (/). Tutti i nomi di percorso IBM WebSphere sono relativi alle directory del sistema dove IBM WebSphere è installato.
ProductDir	Rappresenta la directory in cui viene installato il prodotto. Per l'ambiente IBM WebSphere Business Integration Server Express, la directory predefinita del prodotto è ″IBM\WebSphereICS″.
`%testo%` e *`$testo`*	Un testo che include il simbolo percentuale (%) indica il valore della variabile di sistema `text` di Windows o la variabile utente. La notazione equivalente in un ambiente Linux è *`$testo`*, che indica il valore della variabile *`testo`* in ambiente Linux.

Novità di questa versione

Questa sezione fornisce un sommario delle funzioni nuove o modificate di IBM WebSphere Business Integration Server Express, IBM WebSphere Business Integration Server Express Plus e degli strumenti associati per i gestori di dati. Tali funzioni e gli strumenti ad esse associati sono interamente trattate in questa guida.

Novità nella versione 4.4

Per questa versione, le seguenti modifiche sono state apportate alla guida:

- Il gestore dati Data Complex non è disponibile per l'utilizzo con Linux o OS/400 e i5/OS. Per maggiori dettagli, fare riferimento alle note in Capitolo 9, "Gestore dati Complex Data", a pagina 165.
- Inoltre a partire da questa versione, il gestore dati XML è *globalizzato*. Supporta formati di codifica carattere bidirezionali e l'ODA XML gestisce gli schemi XML e i DTD contenenti questi caratteri BiDi. Le definizioni degli oggetti di business sono memorizzate nel formato di supporto Windows BiDi, CWBF. Tale formato consente agli adattatori di trasformare documenti XML codificati BiDi in oggetti di business e viceversa. Per ulteriori informazioni riguardo all'abilitazione BiDi nel gestore dati XML e l'ODA XML, fare riferimento a "Abilitazione del supporto bidirezionale in ODA XML" a pagina 231.

Novità nel rilascio 4.3.1

Questo manuale non è stato modificato nel rilascio 4.3.1.

Novità del rilascio 4.3

Questo è il primo rilascio di questo manuale.

Parte 1. Guida introduttiva

Capitolo 1. Panoramica del gestore dati

Questo capitolo presenta i gestori di dati del sistema WebSphere Business Integration Server Express. Un gestore dati è responsabile della conversione di oggetti di business in dati serializzati e della conversione di dati serializzati in oggetti di business. Tali dati serializzati sono in formato leggibile dalle applicazioni (stringhe o flussi di dati entranti). Questo capitolo contiene le sezioni seguenti:

Cosa è un gestore dati?

Un *gestore dati* è un'istanza della classe Java che converte tra un particolare formato serializzato e un oggetto di business. I gestori dati sono utilizzati dai componenti di un sistema di integrazione business che trasferisce informazioni tra InterChange Server Express (il broker di integrazione per WebSphere Business Integration Server Express) ed alcuni processi esterni. Tabella 1 mostra i componenti che gestiscono questo trasferimento di informazioni.

Tabella 1. Componenti che trasferiscono informazioni nel sistema WebSphere Business Integration Server Express

Componente	Scopo	Per ulteriori informazioni
Adattatore	Gestisce il trasferimento di informazioni tra InterChange Server Express ed un processo esterno come un'applicazione o una tecnologia. **Nota:** L'adattatore utilizza un componente run-time chiamato *connettore* per gestire realmente il trasferimento di informazioni tra InterChange Server Express ed un'applicazione (o tecnologia). Questi processi esterni identificano gli eventi che accadono al loro interno inviando un record eventi ad un archivio eventi. L'adattatore rileva eventi in un archivio eventi. Quando trova un evento del trigger, l'adattatore crea un oggetto di business che rappresenta l'evento ed invia questo evento *in modo asincrono* a Interchange Server Express. Questo oggetto di business contiene dati e un verbo per indicare il tipo di evento (come Creare o Aggiornare).	Per adattatori IBM: fare riferimento alle guide individuali degli adattatori.

Tabella 1. Componenti che trasferiscono informazioni nel sistema WebSphere Business Integration Server Express (Continua)

Componente	Scopo	Per ulteriori informazioni
Client di accesso	Gestisce il trasferimento di informazioni tra InterChange Server Express e alcuni processi esterni come un servlet all'interno di un server Web. Un client di accesso è un processo esterno che utilizza il Server Access Interface per comunicare direttamente con InterChange Server Express. Quando tale componente riceve alcune informazioni che richiedono di essere trasferite, crea un oggetto di business che rappresenta l'evento e invia questo evento *in modo sincrono* per una collaborazione all'interno di InterChange Server Express. Come con l'adattatore, l'oggetto di business contiene dati e un verbo per indicare il tipo di evento (come Creare o Aggiornare).	Server Access Interface - Guida allo sviluppo

Come Tabella 1 mostra, il compito di entrambi questi componenti (connettore e client di accesso) è di trasferire informazioni tra InterChange Server Express e un processo esterno, come mostrato di seguito:

- Per inviare informazioni a InterChange Server Express, questi componenti formattano le informazioni in un oggetto di business.
- Per inviare informazioni ad un processo esterno, questi componenti le formattano nel loro formato serializzato nativo.

Spesso, il processo esterno utilizza alcuni formati comuni come XML per i suoi dati serializzati nativi. Invece di avere ogni adattatore (o client di accesso) che gestisce la trasformazione tra questi formati comuni e gli oggetti di business, il sistema WebSphere Business Integration Server Express fornisce diversi gestori dati IBM. Inoltre, è possibile creare gestori dati personalizzati per gestire conversioni tra i propri formati nativi. L'adattatore (o il client di accesso) possono quindi chiamare il gestore dati appropriato per effettuare la conversione dati in base al tipo MIME (Multipurpose Internet Mail Extensions) dei dati serializzati.

Nota: Un gestore dati è implementato in una classe Java chiamata `DataHandler`. Tale classe è una classe astratta, che lo sviluppatore del gestore dati estende per implementare un'istanza del gestore dati. Per ulteriori informazioni, consultare "Estensione della classe base del gestore dati" a pagina 179.

Questa sezione fornisce le seguenti informazioni riguardo ai gestori dati:

- "Gestori dati IBM"
- "Meta oggetti del gestore dati" a pagina 6
- "Contesti per gestori dati richiedenti" a pagina 6

Gestori dati IBM

IBM fornisce gestori dati nell'archivio file Java (`jar`) mostrato in Tabella 2. Questi file `jar` risiedono nella sottodirectory `DataHandlers` sotto la directory del prodotto.

Tabella 2. File jar del gestore dati IBM

Contenuto	Descrizione	file jar del gestore dati
Gestori dati base	Gestori dati basati su testo e gestori dati specifici per alcuni adattatori IBM	`CwDataHandler.jar`
Gestori dati speciali	gestore dati XML	`CwXMLDataHandler.jar`
	Gestore dati EDI	`CwEDIDataHandler.jar`
Gestori dati personalizzati	Gestori dati che vengono implementati dall'utente	`CustDataHandler.jar`

Gestori dati base

Il file base del gestore dati, `CwDataHandler.jar`, contiene la maggior parte dei gestori dati IBM. Tale file risiede nella sottodirectory `DataHandlers` della directory del prodotto. Tabella 3 mostra i gestori dati base che questo file base del gestore dati contiene.

Tabella 3. Gestori dati base nel file base del gestore dati

Gestore dati	tipo MIME	Per ulteriori informazioni
Gestore dati Request-Response	`text/requestresponse`	Capitolo 5, "Gestore dati delle richieste-risposte", a pagina 121
Gestore dati FixedWidth	`text/fixedwidth`	Capitolo 6, "gestore dati FixedWidth", a pagina 139
Gestore dati Delimitato	`text/delimited`	Capitolo 7, "Gestore dati delimitato", a pagina 147
Gestore dati NameValue	`text/namevalue`	Capitolo 8, "Gestore dati NameValue", a pagina 155
Gestore dati ContentMaster	diversi	Capitolo 9, "Gestore dati Complex Data", a pagina 165

Nota: Questo manuale descrive i gestori dati di testo che Tabella 3 elenca. Il file base del gestore dati contiene anche diversi gestori dati specifici per alcuni adattatori IBM. Se un adattatore IBM utilizza un gestore dati speciale, la relativa guida adattatore descrive l'installazione, la configurazione e l'utilizzo del relativo gestore dati.

Gestori dati speciali

IBM crea programmi di installazione separati disponibili per pochi gestori dati. Per installare questi gestori dati speciali, è necessario seguire i passaggi forniti nella *Guida per Windows all'installazione di WebSphere Business Integration Server Express* o *per Linux*.

La separazione di un gestore dati dal file base del gestore dati consente alla maggior parte degli adattatori di utilizzare il gestore dati senza incorrere nel sovraccarico di memorizzazione degli altri gestori dati che risiedono nel file base del gestore file. Tabella 4 mostra i gestori dati per cui IBM fornisce programmi di installazione separati e file jar separati.

Tabella 4. Gestori dati IBM con file jar separati

Gestore dati	file jar del gestore dati	tipo MIME	Per ulteriori informazioni
XML Data Handler	`CwXMLDataHandler.jar`	`text/xml`	Capitolo 3, "Gestore dati XML", a pagina 29
Gestore dati EDI	`CwEDIDataHandler.jar`	`edi`	Capitolo 4, "Gestore dati EDI", a pagina 93

Gestori dati personalizzati

Se i gestori dati IBM *non* gestiscono la conversione di dati serializzati in un oggetto di business, è possibile creare il proprio gestore dati personalizzato. Il file`CustDataHandler.jar` è previsto per contenere qualsiasi gestore dati personalizzato che si potrebbe sviluppare. Questo file risiede nella sottodirectory `DataHandlers` della directory del prodotto. Per ulteriori informazioni su come creare un gestore dati personalizzato, fare riferimento a Capitolo 10, “Configurazione di un gestore dati personalizzato”, a pagina 175.

Nota: Per fornire assistenza nello sviluppo di gestori dati personalizzati, IBM fornisce anche il codice sorgente per i gestori dati FixedWidth, Delimited e NameValue come codice di esempio. Per ulteriori informazioni, consultare “Esempi gestori dati” a pagina 177.

Meta oggetti del gestore dati

Un connettore o un processo Server Access Interface istanziano il gestore dati in base al tipo MIME di un file di immissione o il tipo MIME specificato in una richiesta oggetto business.

Un meta oggetto del gestore dati è un oggetto di business gerarchico che può contenere qualsiasi numero di oggetti secondari. Le informazioni di configurazione del gestore dati sono organizzate nella seguente gerarchia:

- Il meta oggetto *di livello superiore* contiene informazioni sui tipi MIME che i differenti gestori dati possono supportare. Ogni attributo di livello superiore è un attributo a cardinalità 1 che assegna un meta oggetto secondario per un'istanza del gestore dati. Ogni attributo rappresenta un tipo MIME ed indica quale gestore dati può maneggiarlo.
- Il *secondario* meta oggetto contiene le informazioni di configurazione effettive per un particolare gestore dati. Ogni attributo rappresenta una proprietà di configurazione e fornisce informazioni come il relativo valore predefinito e il tipo.

Nota: Un gestore dati non è richiesto per utilizzare i meta oggetti per contenere le informazioni di configurazione. Tuttavia, *tutti* i gestori dati IBM sono progettati per utilizzare i meta oggetti per le loro informazioni di configurazione.

I meta oggetti del gestore dati consentono ad un connettore o al processo Server Access Interface di istanziare il gestore dati basato sul tipo MIME di un file di immissione o il tipo MIME specificato in una richiesta oggetto business. Per configurare un gestore dati, assicurarsi che i suoi meta-oggetti siano correttamente inizializzati e disponibili per i richiedenti (un connettore o un client di accesso).

Nota: Ciascun gestore dati IBM utilizza proprietà di configurazione che sono definite nei meta-oggetti del gestore dati. Tuttavia, un gestore dati personalizzato può o non può utilizzare meta-oggetti per le sue proprietà di configurazione. Per ulteriori informazioni, consultare “Utilizzo dei meta-oggetti del gestore dati” a pagina 179.

Contesti per gestori dati richiedenti

Come Tabella 1 a pagina 3 descrive, un componente che necessita di trasferire dati nel sistema WebSphere Business Integration Server Express può richiamare un gestore dati. Tabella 5 fornisce informazioni aggiuntive sui componenti che un gestore dati può richiamare.

Tabella 5. Contesti per gestori dati richiedenti

Componente	Tipo di comunicazione eventi	Tipo di flusso	Software che richiama il gestore dati
Adattatore	Asincrono	Flusso attivazione eventi	Connettore
Client di accesso	Sincrona	Flusso attivato da chiamata	Server Access Interface

Come Tabella 5 mostra, in un flusso di attivazione eventi, un adattatore chiama un gestore dati direttamente. In un flusso di attivazione di chiamata, un processo esterno che utilizza Server Access Interface (chiamato un client di accesso) dà inizio ad una chiamata al gestore dati. Un gestore dati funziona allo stesso modo se viene chiamato direttamente da un adattatore o indirettamente da un client di accesso. Questi contesti vengono descritti nelle prossime sezioni.

Gestori dati in un contesto connettore

In un flusso di attivazione eventi, il componente di esecuzione di un adattatore, chiamato il *connettore*, interagisce direttamente con un gestore dati per convertire i dati.

Nota: Per gli adattatori IBM, fare riferimento alle guide individuali degli adattatori. In base alla lingua in cui l'adattatore è implementato.

Quando un connettore chiama un gestore dati, il gestore dati viene eseguito come parte del processo del connettore. Figura 1 illustra il gestore dati nel contesto di un connettore.

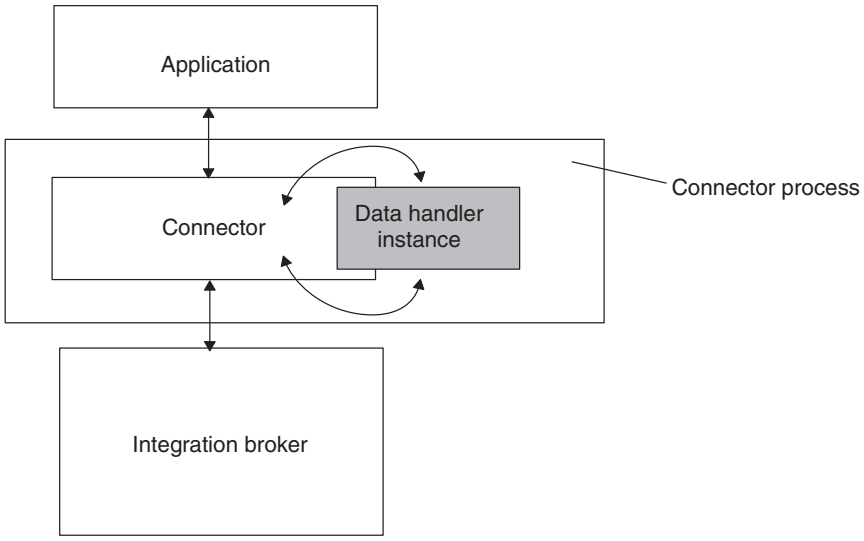

Figura 1. Gestore dati nel contesto di un connettore

LA conversione dati riporta i requisiti dell'oggetto di business e la direzione del flusso:

- Il connettore richiama il gestore dati per una conversione oggetto di business in stringa quando gestisce richieste di elaborazione di un oggetto di business.
- Il connettore richiama il gestore dati per una conversione stringa in oggetto di business quando gestisce una notifica di eventi.

Connettore conversione oggetto di business in stringa: Per una conversione oggetto di business in stringa, il connettore chiama il gestore dati, passando un oggetto di business. Il gestore dati utilizza le informazioni nell'oggetto di business e la definizione di oggetto di business per creare un flusso o una stringa di dati. Questo flusso o stringa di dati è nel formato associato con il gestore dati, generalmente di un particolare tipo MIME. La conversione oggetto di business in stringa è utile quando il connettore riceve informazioni da InterChange Server Express nella forma di un oggetto di business. Il connettore deve quindi inviare le informazioni nell'oggetto di business alle relative applicazioni (o tecnologia) come dati serializzati.

Figura 2 illustra il gestore dati nel contesto di un connettore quando il gestore dati effettua una conversione oggetto di business in stringa.

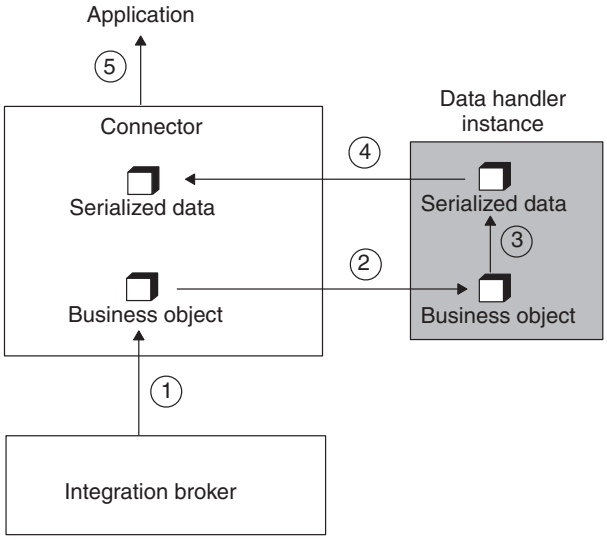

Figura 2. Conversione oggetto di business in stringa nel contesto del connettore

1. Il connettore riceve un oggetto di business da InterChange Server Express.
2. Il connettore crea un'istanza di un gestore dati per gestire l'oggetto di business (utilizzando il metodo statico `createHandler()` nella classe base `DataHandler`).

   Per ulteriori informazioni riguardo a come un connettore istanzia il gestore dati, fare riferimento a "Avvio di un'istanza nel contesto di un connettore" a pagina 17.
3. Il connettore richiede una conversione oggetto di business in stringa chiamando uno dei seguenti metodi del gestore dati:
   - `getStreamFromBO()`
   - `getStringFromBO()`
   - `getByteArrayFromBO()`

   In questo metodo, il connettore invia l'oggetto di business come un argomento. Il gestore dati serializza l'oggetto di business object nel formato dati richiesto.
4. Il gestore dati restituisce i dati serializzati al connettore.
5. IL connettore scrive i dati serializzati nella destinazione, che può essere un e-mail, un file o una connessione HTTP.

Conversione connettore stringa in oggetto di business: Per una conversione stringa in oggetto di business, il connettore chiama il gestore dati, passando i dati serializzati il relativo oggetto di tipo MIME associato. Il gestore dati riceve un

flusso o una stringa di dati. Il gestore dati utilizza le informazioni nel flusso di dati per creare, denominare e compilare un'istanza di oggetto di business del tipo specificato. La conversione stringa in oggetto di business è utile quando il connettore ha bisogno di inviare un evento a InterChange Server Express. L'applicazione invia questo evento come dato serializzato che ha un particolare tipo MIME, al connettore.

Figura 3 illustra il gestore dati nel contesto di un connettore quando il gestore dati effettua una conversione stringa in oggetto di business.

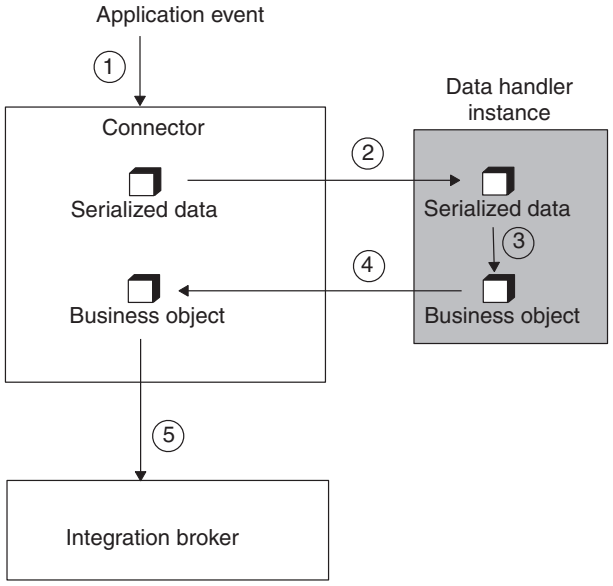

Figura 3. Conversione stringa in oggetto di business nel contesto del connettore

1. Il connettore rileva un evento applicazione. L'evento potrebbe essere nella forma di un e-mail, un file di testo, un documento XML o qualsiasi altro formato comune per cui il gestore dati esiste.
2. Il connettore crea un'istanza di un gestore dati per gestire l'evento (utilizzando il metodo statico `createHandler()` nella classe base `DataHandler`).

   Per ulteriori informazioni riguardo a come un connettore istanzia il gestore dati, fare riferimento a "Avvio di un'istanza nel contesto di un connettore" a pagina 17.
3. Il connettore invia nei dati serializzati come un argomento al metodo `getBO()` dell'istanza del gestore dati. Il gestore dati crea un'istanza di un oggetto di business.

   Il connettore potrebbe anche specificare l'oggetto di business in cui il metodo `getBO()` converte i dati. Alcuni connettori specificano il tipo di oggetto di business; altri presumono che il gestore dati può estrarre il tipo dell'oggetto di business dal testo serializzato. Il gestore dati analizza i dati e compila i valori dell'attributo per l'oggetto di business in base ai dati serializzati.
4. Il gestore dati restituisce l'oggetto di business al connettore.
5. Il connettore invia l'oggetto di business a InterChange Server Express.

I gestori dati nel contesto di Server Access Interface

In un flusso di attivazione di chiamata, un client di accesso interagisce con un gestore dati per convertire i dati. Un *client di accesso* è un processo esterno che utilizza Server Access Interface per interagisce con InterChange Server Express.

Quando un client di accesso chiama o il metodo `ItoExternalForm()` o `IcreateBusinessObjectFrom()` di Server Access Interface API, avvia un chiamata verso un gestore dati. Server Access Interface, che viene eseguito come parte del processo InterChange Server Express, richiama il gestore dati.

InterChange Server Express

Server Access Interface è un API che consente ad un client di accesso di eseguire una collaborazione in InterChange Server Express. Quest'interfaccia è disponibile per l'utilizzo *solo* quando InterChange Server Express è il broker di integrazione. Per ulteriori informazioni su questa interfaccia e sui client di accesso, fare riferimento a *guida allo sviluppo Access* nella IBM WebSphere Business Integration Server Express serie di documentazioni.

Un client di accesso potrebbe essere un servlet che gestisce una richiesta da un client browser. La richiesta può essere una richiesta di dati, una richiesta di ordine o un altro tipo di transazione business a business. Come altro esempio, un client di accesso può essere un programma C++ o Java che utilizza Server Access Interface per accedere a InterChange Server Express e scambiare dati con un'altra applicazione.

Quando un client di accesso inizia una chiamata che richiede un gestore dati, il gestore dati viene eseguito come parte del processo di InterChange Server Express. Figura 4 illustra il gestore dati nel contesto di Server Access Interface. In questo esempio, il client di accesso è un server Web e un servlet.

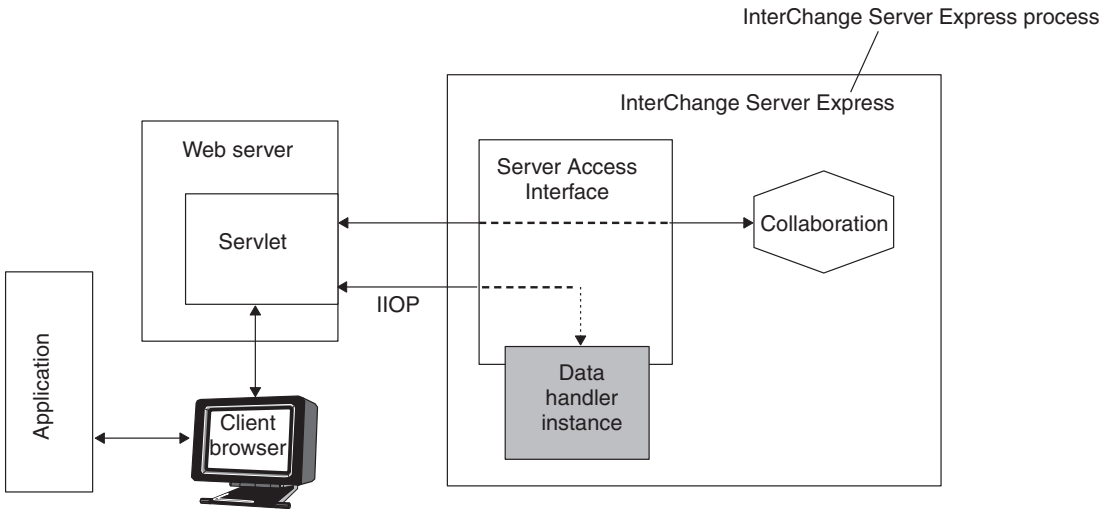

Figura 4. Gestore dati nel contesto di Server Access Interface

La conversione dati riporta i requisiti dell'oggetto di business e la direzione del flusso:

- Ricevendo un oggetto di business da InterChange Server Express richiede che il client di accesso interroghi il gestore dati per una conversione oggetto di business in stringa.
- Inviando dati a InterChange Server Express richiede che il client di accesso interroghi il gestore dati per una conversione stringa in oggetto di business.

Conversione oggetto di business in stringa di Server Access Interface: Per una conversione oggetto di business in stringa, il gestore dati riceve un oggetto di business come risultato di una esecuzione di una collaborazione. Il gestore dati utilizza le informazioni nell'oggetto di business per creare un flusso o una stringa di dati. Questi dati sono nel formato associato con il gestore dati, in genere di un particolare tipo MIME. Il client di accesso spesso invia l'oggetto di business risultante all'applicazione come dati serializzati.

Figura 5 illustra il gestore dati nel contesto di Server Access Interface quando il gestore dati effettua una conversione oggetto di business in stringa per un client di accesso.

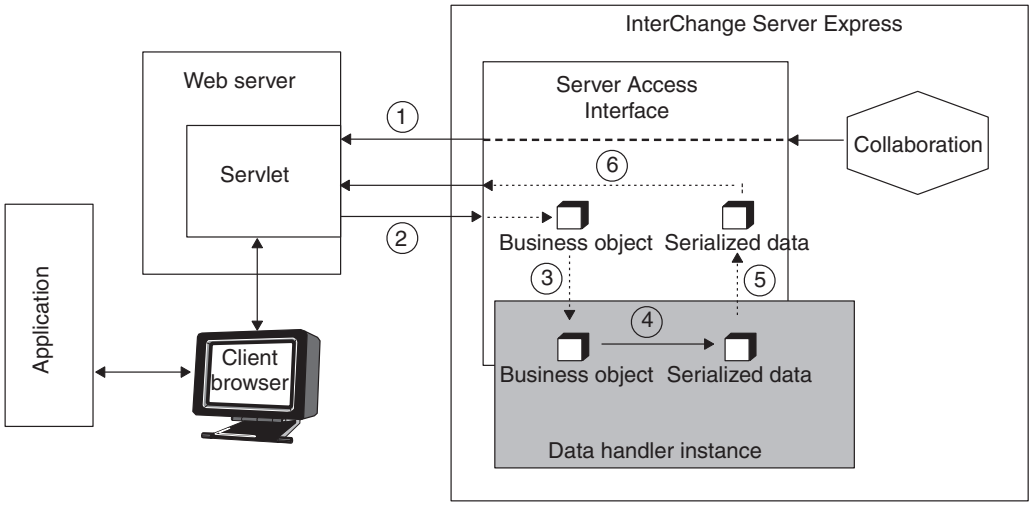

Figura 5. Conversione oggetto di business in stringa nel contesto di Server Access Interface

1. La collaborazione restituisce i dati richiesti o i risultati delle azioni richieste al client di accesso.
2. Per convertire l'oggetto di business nel formato richiesto, il client di accesso invia l'oggetto di business come un argomento al metodo `ItoExternalForm()` di Server Access Interface.
3. Server Access Interface effettua le seguenti operazioni:
   - Creazione di un'istanza del gestore dati per gestire la conversione (utilizzando il metodo statico `createHandler()` nella classe base `DataHandler`).
   - Invio nei dati serializzati come un argomento ad uno dei seguenti metodi del gestore dati:
     - `getStreamFromBO()`
     - `getStringFromBO()`
     - `getByteArrayFromBO()`
4. Il gestore dati analizza l'oggetto di business per creare i dati serializzati.
5. Il gestore dati restituisce i dati serializzati a Server Access Interface.
6. Server Access Interface restituisce i dati serializzati al client di accesso.

Conversione stringa in oggetto di business di Server Access Interface: Per una conversione stringa in oggetto di business, il gestore dati riceve un flusso o una stringa di dati. Il gestore dati utilizza le informazioni nel flusso di dati per creare, denominare e compilare un'istanza di oggetto di business del tipo specificato. La conversione string in oggetto di business è utile quando il client di accesso ha

bisogno di inviare un oggetto di business per una collaborazione in InterChange Server Express. Il client di accesso invia i dati serializzati, in genere che hanno un particolare tipo MIME, al gestore dati.

Figura 6 illustra il gestore dati nel contesto di Server Access Interface quando il gestore dati effettua una conversione stringa in oggetto di business per un client di accesso.

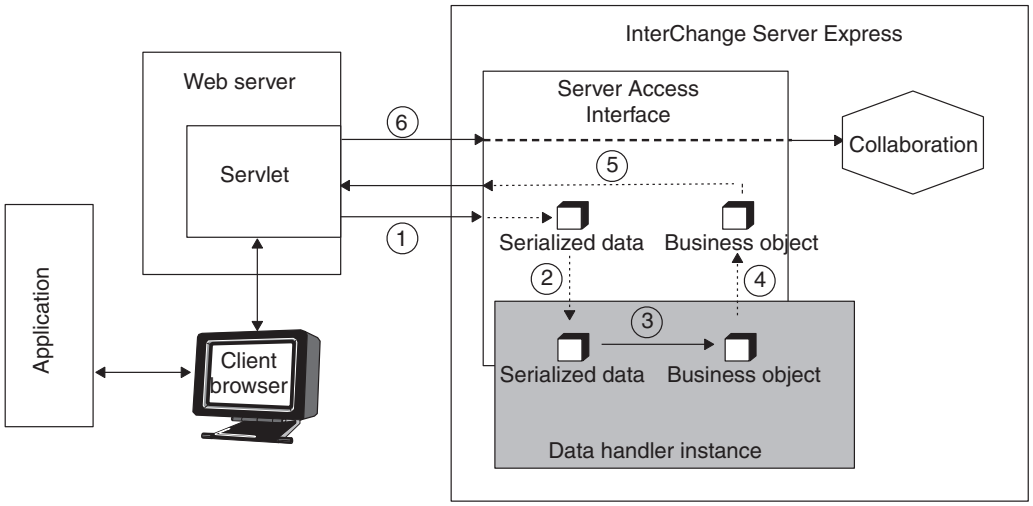

Figura 6. Conversione stringa oggetto di business nel contesto di Server Access Interface

1. Per convertire i dati serializzati in un oggetto di business, il client di accesso invia i dati serializzati come un argomento al metodo `IcreateBusinessObjectFrom()` di Server Access Interface.
2. Server Access Interface effettua le seguenti operazioni:
   - Creazione di un'istanza del gestore dati per gestire la conversione (utilizzando il metodo statico `createHandler()` nella classe base `DataHandler`).
   - Invio nei dati serializzati come un argomento al metodo `getBO()` dell'istanza del gestore dati.
3. Il gestore dati crea un'istanza di un oggetto di business.

   Il gestore dati analizza i dati e compila i valori dell'attributo per l'oggetto di business in base ai dati serializzati.
4. Il gestore dati restituisce l'oggetto di business a Server Access Interface.
5. Server Access Interface restituisce l'oggetto di business al client di accesso.
6. Il client di accesso richiama una collaborazione che utilizzi i dati dell'oggetto in un processo di business.

Istanziare il gestore dati

I gestori dati sono implementati come una libreria di classi che un connettore o the Server Access Interface (per un client di accesso che comunica con InterChange Server Express) possono utilizzare. La classe base `DataHandler` è una classe astratta. Perciò, per istanziare un gestore dati, è necessario istanziare una delle sottoclassi `DataHandler`. Ogni gestore dati, sia un gestore dati IBM che un gestore dati personalizzato, è una sottoclasse della classe base `DataHandler`. Il metodo per istanziare un'istanza di un gestore dati è `createHandler()`.

Il metodo `createHandler()` utilizza le informazioni nei *meta oggetti del gestore dati* per determinare quale gestore dati istanziare e come inizializzare questo gestore dati. Un meta oggetto del gestore dati è un oggetto di business gerarchico che può contenere qualsiasi numero di oggetti secondari. Le informazioni di configurazione del gestore dati sono organizzate nella seguente gerarchia:

- Il meta oggetto *di livello superiore* contiene informazioni sui tipi MIME che i differenti gestori dati possono supportare. Ogni attributo di livello superiore è un attributo a cardinalità 1 che assegna un meta oggetto secondario per un'istanza del gestore dati. Ogni attributo rappresenta un tipo MIME e il relativo tipo attributo indica il meta oggetto secondario per il gestore dati che può utilizzare questo tipo MIME.
- Il meta oggetto *secondario* contiene le informazioni di configurazione effettive per un particolare gestore dati. Ogni attributo rappresenta una proprietà di configurazione e fornisce informazioni come il relativo valore predefinito e il tipo.

Nota: Un gestore dati non è richiesto per utilizzare i meta oggetti per contenere le informazioni di configurazione. Tuttavia, *tutti* i gestori dati IBM sono progettati per utilizzare i meta oggetti per le loro informazioni di configurazione.

Il metodo `createHandler()` effettua i seguenti passaggi per istanziare un gestore dati:

- "Identificazione della classe del gestore dati"
- "Impostazione delle proprietà di configurazione" a pagina 15
- "Impostazione del prefisso dell'oggetto di business" a pagina 16

Identificazione della classe del gestore dati

Perché un gestore dati sia creato,deve essere istanziata un'implementazione della classe base `DataHandler`. Il metodo di creazione di un'istanza del gestore dati deriva il nome di questa classe del gestore dati da uno di due valori, che sono passati come un argomento al metodo `createHandler()`:

- Il nome classe del gestore dati da istanziare
- Il tipo MIME dei dati da convertire

Utilizzo di un nome classe

Se il gestore dati richiedente passa in un nome classe come un argomento, `createHandler()` istanzia un gestore dati di quel nome classe. Cerca la classe specificata nei seguenti posti:

1. `CwDataHandler.jar` file
2. `CwXMLDataHandler.jar` file
3. `CwEDIDataHandler.jar` file
4. `CustDataHandler.jar` file
5. Altrove in `CLASSPATH`

Se il gestore dati richiedente fornisce *solo* il nome classe per il gestore dati, `createHandler()` non cerca i meta oggetti del gestore dati né imposta le proprietà di configurazione da questi oggetti. Perciò, un gestore dati istanziato in questo modo *non* richiede meta oggetti. Per ulteriori informazioni se un gestore personalizzato dovrebbe utilizzare i meta oggetti, fare riferimento a "Utilizzo dei meta-oggetti del gestore dati" a pagina 179.

Utilizzo di un tipo MIME

Se il gestore dati richiedente *non* passa il nome classe come un argomento, il metodo `createHandler()` richiede un valore per un tipo MIME. Quando un componente richiedente (connettore o client di accesso) passa un tipo MIME, `createHandler()` utilizza il meta oggetto secondario del gestore dati associato con questo tipo MIME per derivare il nome classe ed altre informazioni di configurazione per l'istanza del gestore dati.

Nota: Per ulteriori informazioni sui meta oggetti, fare riferimento a "Configurazione dei gestori dati" a pagina 21. Per ulteriori informazioni se un gestore personalizzato dovrebbe utilizzare i meta oggetti, fare riferimento a "Utilizzo dei meta-oggetti del gestore dati" a pagina 179.

Per derivare un nome classe da un tipo MIME specificato, il metodo `createHandler()` effettua i seguenti passaggi:

1. Conversione del tipo MIME in una stringa del tipo MIME

   Quando il metodo `createHandler()` ricerca il meta oggetto di livello superiore del gestore dati, converte ogni carattere non alfanumerico, come il trattino (-), il punto (.) o la barra (/), in un carattere sottolineatura (_). Ad esempio, se il tipo MIME è `text/html`, `createHandler()` analizza il tipo nella stringa `text_html`.

   Il metodo `createHandler()` effettua tale conversione di caratteri non alfanumerici in fasi, in modo che le corrispondenze dei nomi del tipo MIME che contiene punti possano avvenire. Tuttavia, Business Object Designer Express non consente il carattere punto nei nomi attributo. Perciò, IBM raccomanda di *non* includerli nei nomi del tipo MIME.

   E' possibile creare combinazioni uniche tipo MIME/sottotipo per indicare variazioni di un particolare tipo MIME. Nel nome del tipo MIME, è possibile separare il tipo MIME e il sottotipo con un carattere non alfanumerico (ad esempio un trattino o un carattere sottolineatura). Tuttavia, poiché `createHandler()` sostituisce qualsiasi carattere non alfanumerico con un carattere sottolineatura, IBM raccomanda di utilizzare solo un carattere sottolineatura per separare il tipo MIME e il sottotipo. Se il tipo MIME è `text/xml-sgml`, il metodo converte il tipo nella stringa `text_xml_sgml`.
2. Ottenere il nome del meta oggetto di livello superiore del gestore dati da una proprietà statica nella classe base `DataHandler`. In questo meta oggetto di livello superiore del gestore dati, `createHandler()` ricerca un attributo che corrisponda alla stringa del tipo MIME dei dati da convertire.
3. Se `createHandler()` localizza una stringa corrispondente del tipo MIME nel meta oggetto di livello superiore, `createHandler()` effettua i seguenti passaggi:
   a. Se il richiedente ha fornito un valore per il prefisso dell'oggetto (un terzo argomento facoltativo) per `createHandler()`, interpreterà questo valore per un sottotipo MIME e lo aggiunge al tipo MIME per creare una stringa del tipo MIME del formato:

      MIMETypeString_BOPrefix

      Se non esiste alcun attributo di quest nome, `createHandler()` ricerca un attributo che corrisponda *solo* al tipo MIME. Ad esempio, se il richiedente passa in un tipo MIME di `edi` e un prefisso dell'oggetto di business di `x12`, `createHandler()` ricerca un attributo nel meta oggetto di livello superiore denominato "`edi_x12`". Se non esiste alcun attributo di questo nome, `createHandler()` ricerca un attributo denominato "`edi`".
   b. Ottenere il meta oggetto secondario associato del gestore dati, che contiene le proprietà di configurazione per l'istanza del gestore dati da utilizzare (ad

esempio, una possibile proprietà di configurazione può identificare quale carattere utilizzare per il delimitatore del gestore dati).

c. Ottenere il valore dell'attributo `ClassName` dal meta oggetto secondario.
   - Se l'attributo `ClassName` contiene un valore, `createHandler()` istanzia un gestore dati di questo nome classe. Parte del processo di implementazione di un gestore dati è la creazione del meta oggetto secondario associato del gestore dati. A questo punto, lo sviluppatore che implementa il gestore dati può aggiungere il nome classe all'attributo `ClassName` del meta oggetto secondario. Tale valore `ClassName` viene richiesto se il nome classe è differente dal nome classe predefinito, come descritto in 4a.
   - Se questo attributo `ClassName` *non* contiene valori, `createHandler()` crea un nome classe per il gestore dati che istanzia, come descritto in 4a.

4. Se il metodo `createHandler()` *non* trova una stringa del tipo MIME corrispondente nel meta oggetto di livello superiore, crea un nome classe del gestore dati che dovrà istanziare nel seguente modo:
   a. Aggiungere la stringa del tipo MIME al nome del pacchetto base del gestore dati:

      ```
      com.crossworlds.DataHandlers
      ```

      Ad esempio, se la stringa del tipo MIME è `text_html`, la stringa risultante sarà:

      ```
      com.crossworlds.DataHandlers.text.html
      ```

   b. Se il metodo può localizzare il nome classe generato su CLASSPATH, istanzierà tale classe. Ricercherà tale classe nei seguenti posti:
      - `CwDataHandler.jar` file
      - `CustDataHandler.jar` file
   - Altrove in CLASSPATH

Utilizzo di un nome classe e tipo MIME

Se il richiedente fornisce *sia* il nome classe che il tipo MIME , `createHandler()` effettua le seguenti operazioni:

- Creare un gestore dati di una classe specifica. Per localizzare tale classe, il gestore dati ricerca nelle directory elencate in "Utilizzo di un nome classe" a pagina 13.
- Utilizzo del meta oggetto secondario associato col tipo MIME per inizializzare le opzioni di configurazione di questo gestore dati. Per informazioni su come `createHandler()` determina il meta oggetto secondario dal tipo MIME, fare riferimento a "Utilizzo di un tipo MIME" a pagina 14

In altre parole, quando il richiedente fornisce un nome classe, questo nome classe *ignora* l nome classe specificato nell'attributo `ClassName` del meta oggetto secondario.

Impostazione delle proprietà di configurazione

Tutti i gestori dati IBM (fare riferimento a Tabella 3 e Tabella 4) sono progettati per utilizzare i meta oggetti del gestore dati per le loro informazioni di configurazione. Un meta oggetto del gestore dati è un oggetto di business gerarchico:

- Nel meta oggetto di livello superiore del gestore dati, ogni attributo viene identificato con un tipo MIME e rappresenta un meta oggetto secondario del gestore dati.
- Il meta oggetto secondario del gestore dati contiene informazioni di configurazione per il gestore dati associato con il tipo MIME.

I gestori dati IBM utilizzano le informazioni di configurazione nel relativo meta oggetto secondario associato del gestore dati per inizializzare le sue proprietà. Perciò, IBM fornisce un meta oggetto secondario per ogni relativo gestore dati fornito (fare riferimento a Tabella 8).

Nota: Per una descrizione approfondita dei meta oggetti del gestore dati, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

Dopo che il metodo `createHandler()` istanzia l'istanza del gestore dati, richiama uno speciale metodo protetto, `setupOptions()`, per inizializzare la configurazione del gestore dati con i valori nell'appropriato meta oggetto secondario del gestore dati.

Nota: Un gestore dati personalizzato non è richiesto per utilizzare i meta oggetti ad inizializzare la relativa configurazione. Se il gestore dati non presenta un meta oggetto secondario associato, il metodo `createHandler()` non richiama `setupOptions()` per il gestore dati. Per ulteriori informazioni, consultare "Impostazione dei altri oggetti di business" a pagina 198.

Per ulteriori informazioni sui meta oggetti, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

Impostazione del prefisso dell'oggetto di business

E' possibile che il metodo `createHandler()` accetti un prefisso facoltativo dell'oggetto di business come suo terzo argomento. Questo argomento viene utilizzato per determinare il nome del tipo MIME (fare riferimento a "Utilizzo di un tipo MIME" a pagina 14). Dopo che `createHandler()` ha istanziato l'istanza del gestore dati ed impostato le relative proprietà di configurazione, la sua ultima operazione è di impostare il valore dell'opzione di configurazione `BOPrefix` (se ne esiste una) nel gestore dati sul valore di questo terzo argomento.

Nota: Qualsiasi gestore dati che utilizzi un'opzione di configurazione `BOPrefix` (come il gestore dati XML) preaggiunge questo prefisso ai nomi dell'oggetto di business.

Il gestore dati può aggiungere questo prefisso ai nomi di qualsiasi oggetto di business che crea (durante una conversione stringa in oggetto di business). Aggiunge un carattere sottolineatura (_) tra il prefisso e il nome dell'oggetto di business. Ad esempio, un connettore può richiamare il gestore dati XML con la seguente chiamata `createHandler()`:

```
createHandler(null, "text/xml", "UserApp");
```

Il metodo `createHandler()` istanzia il gestore dati XML and imposta il relativo attributo `BOPrefix` su "`UserApp`". Quando il gestore dati XML crea un oggetto di business `Customer`, l'oggetto di business:

- Ottiene il nome dell'oggetto di business dai dati serializzati
- Aggiunge il prefisso "`UserApp`" al nome dell'oggetto di business

Il risultante nome dell'oggetto di business è "`UserApp_Customer`".

Richiama il gestore dati

A prescindere dal contesto in cui un gestore dati viene richiamato, il gestore dati viene istanziato dal metodo `createHandler()`. Il modo in cui il metodo viene richiamato in ogni contesto è il seguente:

- Un connettore *esplicitamente* richiama `createHandler()` per istanziare un gestore dati.
- Un client di accesso richiama `createHandler()` *implicitamente*; Server Access Interface chiama effettivamente `createHandler()` quando il client di accesso avvia una chiamata del gestore dati attraverso uno dei seguenti metodi di Server Access Interface: `ItoExternalForm()` o `IcreateBusinessObjectFrom()`.

Avvio di un'istanza nel contesto di un connettore

Figura 7 mostra un esempio di un avvio di un'istanza del gestore dati quando il gestore dati viene richiamato nel contesto del connettore.

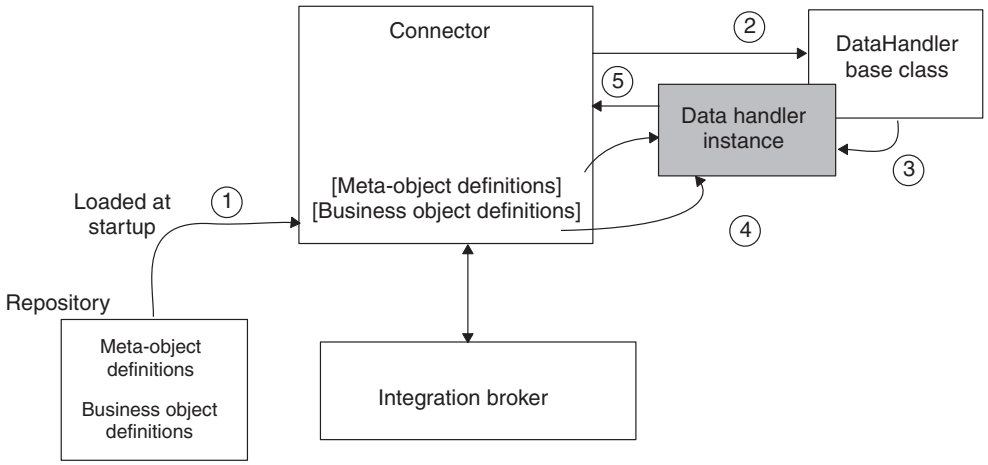

Figura 7. Avvio di un'istanza del gestore dati nel contesto di un connettore

Per istanziare un gestore dati richiamato nel contesto di un connettore, il connettore effettua i seguenti passaggi:

1. I meta oggetti vengono letti nella memoria quando il connettore viene avviato, sempre che i meta oggetti siano inclusi nell'elenco degli oggetti supportati da un connettore.

   I meta oggetti sono conservati nel repository insieme agli oggetti di business. Le definizioni del meta oggetto, come le definizioni dell'oggetto di business, devono esistere nella memoria per il gestore dati per essere in grado di accedere ad essi. Quando questi meta oggetti sono nella memoria come parte del processo del connettore, i meta oggetti diventano accessibili per un gestore dati richiamato nel contesto di un connettore.
2. Il connettore richiama il metodo statico `setConfigMOName()` nella classe base `DataHandler` per impostare una proprietà statica nella classe base del gestore dati al nome del livello superiore meta oggetto per il gestore dati.

   Questo meta oggetto di livello superiore è il meta oggetto del connettore (`MO_DataHandler_Default` per impostazione predefinita). Il meta oggetto di livello superiore deve essere parte dell'elenco degli oggetti supportati del connettore.

   Nota: Come il connettore ottiene il nome del meta oggetto di livello superiore del gestore dati viene determinato come parte del design del connettore. Per ulteriori informazioni, consultare "Configurazione di un connettore" a pagina 199.

3. Il connettore richiama il metodo statico `createHandler()` nella classe base `DataHandler` per creare un'istanza della classe base `DataHandler` che effettua la conversione dei dati richiesti. Il nome della classe da creare viene determinato in questi due modi:
   - Se il connettore passa in un nome classe come un argomento, il metodo `createHandler()` istanzia un gestore dati di quel nome classe. Un connettore può esplicitamente specificare un nome classe quando richiama `createHandler()`.
   - Se il tipo MIME viene passato invece del nome classe, `createHandler()` deriva il nome classe dal tipo MIME.

     Il metodo `createHandler()` converte il tipo MIME in una stringa del tipo MIME e ottiene il nome del meta oggetto di livello superiore del gestore dati da una proprietà statica nella classe base `DataHandler`. Da questo meta oggetto di livello superiore, `createHandler()` ottiene il nome del meta oggetto secondario per il gestore dati. Questo meta oggetto secondario contiene informazioni di configurazione, incluso il nome della classe da istanziare. Per ulteriori informazioni su come questa derivazione avvenga, fare riferimento "Identificazione della classe del gestore dati" a pagina 13.
4. Il gestore dati effettua la conversione dei dati richiesti. L'agente del connettore richiama i metodi appropriati `DataHandler` per effettuare la conversione richiesta:
   - Il metodo `getBO()` per una conversione stringa in oggetto di business.
   - Il metodo `getStringFromBO()` per una conversione oggetto di business in stringa o il `getStreamFromBO()` metodo per una conversione oggetto di business in flusso.
5. Il gestore dati restituisce il formato appropriato all'agente del connettore.

Avvio di un'istanza nel contesto di Server Access Interface

Figura 8 mostra un esempio di un avvio di un'istanza del gestore dati quando il gestore dati viene richiamato nel contesto di Server Access Interface.

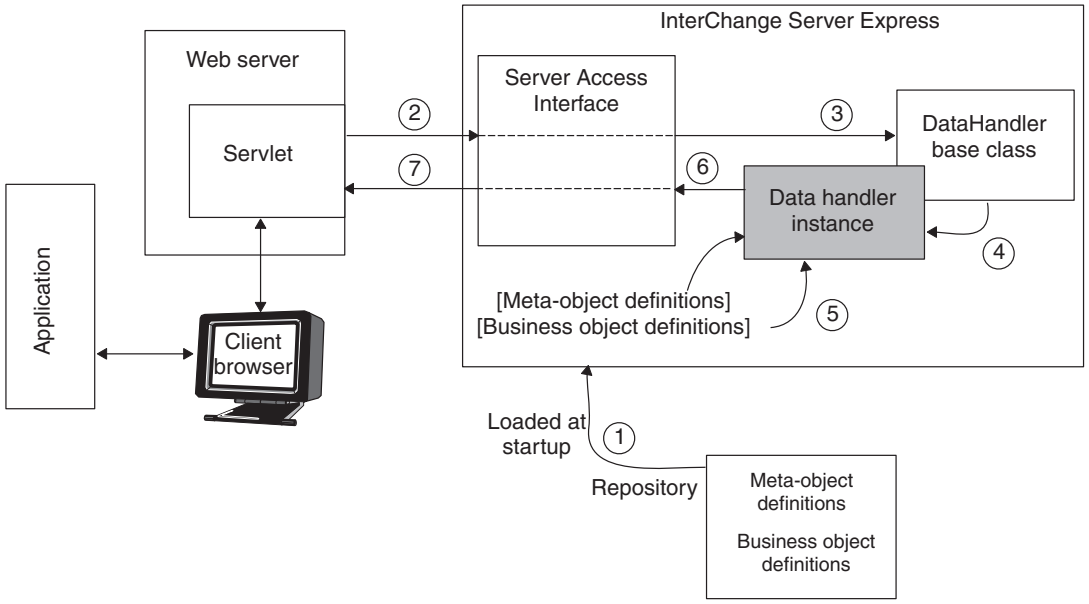

Figura 8. Avvio di un'istanza del gestore dati nel contesto di Server Access Interface

Per istanziare un gestore dati richiamato nel contesto di Server Access Interface, Server Access Interface effettua i seguenti passaggi:

1. I meta oggetti vengono letti nella memoria quando il server viene avviato, insieme con tutte le altre definizioni dell'oggetto di business nel repository.

   I meta oggetti sono immagazzinati nel repository insieme con gli oggetti di business. Le definizioni del meta oggetto, come le definizioni dell'oggetto di business, devono esistere nella memoria per il gestore dati per essere in grado di accedere ad essi. Una volta che questi meta oggetti sono nella memoria come parte del processo InterChange Server Express (e Server Access Interface), i meta oggetti diventano accessibili per un gestore dati richiamato nel contesto di Server Access Interface.
2. Un client di accesso avvia la creazione di un'istanza di un gestore dati con uno dei metodi di Server Access Interface : `IcreateBusinessObjectFrom()` o `ItoExternalForm()`.

   Questi metodi passano il tipo MIME dei dati che devono essere convertiti.

   Nota: E' necessario che i client di accesso utilizzino i metodi di Server Access Interface per richiamare un gestore dati. Sebbene questi metodi richiamino indirettamente i metodi interfaccia del gestore dati, i metodi forniscono solo un sottoinsieme dell'interfaccia del gestore dati. Per ulteriori informazioni sui metodi di Server Access Interface , fare riferimento a *Guida allo sviluppo Access*. Per ulteriori informazioni sui metodi forniti nell'interfaccia del gestore dati, fare riferimento a Capitolo 11, "Metodi della classe base del gestore dati", a pagina 203.
3. Server Access Interface imposta il nome del meta oggetto di livello superiore per il gestore dati da `MO_Server_DataHandler`.
4. Server Access Interface crea un'istanza di una sottoclasse `DataHandler` per effettuare le conversioni di dati richieste (utilizzando il metodo `createHandler()` della classe base `DataHandler`).

   Una volta richiamato nel contesto di Server Access Interface, il metodo `createHandler()` *non* specifica un nome classe. Invece, `createHandler()` converte il tipo MIME in una stringa del tipo MIME ed ottiene il nome del meta oggetto di livello superiore del gestore dati. Da questo meta oggetto di livello superiore, `createHandler()` ottiene il nome del meta oggetto secondario per il gestore dati. Questo meta oggetto secondario contiene informazioni di configurazione, incluso il nome della classe da istanziare. Per ulteriori informazioni su come questa derivazione avvenga, fare riferimento "Identificazione della classe del gestore dati" a pagina 13.
5. Il gestore dati effettua la conversione dei dati richiesti. Server Access Interface chiama il metodo `DataHandler` appropriato per effettuare la conversione richiesta:
   - Il metodo `getBO()` per una conversione stringa in oggetto di business
   - Il metodo `getStringFromBO()` per una conversione oggetto di business in stringa o il `getStreamFromBO()` metodo per una conversione oggetto di business in flusso.
6. Il gestore dati restituisce il formato richiesto a Server Access Interface.
7. Server Access Interface restituisce il formato richiesto al client di accesso.

Progettazione del gestore dati con i metadati

I gestori dati IBM presentano metadati. I metadati sono dati che riguardano l'oggetti di business immagazzinato nelle definizioni dell'oggetto di business. I metadati in una definizione di un oggetto di business forniscono informazioni che descrivono i dati in un'istanza dell'oggetto di business. In generale, i metadati dell'oggetto di business includono la struttura dell'oggetto di business, le impostazioni delle relative proprietà dell'attributo e il contenuto delle relative informazioni specifiche sull'applicazione. Inoltre, forniscono istruzioni su come elaborare i dati.

I connettori sono generalmente progettati per utilizzare i metadati dell'oggetto di business quando elaborano gli oggetti di business. Allo stesso modo, i gestori dati sono progettati per utilizzare i metadati dell'oggetto di business. Ad esempio:

- Il gestore dati XML richiede che ogni definizione dell'oggetto di business contenga informazioni specifiche sull'applicazione che descrivano ogni attributo. Questo testo abilita il gestore dati ad identificare gli elementi XML, gli attributi XML, le istruzioni di elaborazione ed altri tipi di markup XML.
- Il gestore dati FixedWidth utilizza il valore dell'attributo della proprietà MaxLenght dell'oggetto di business per analizzare le stringhe a grandezza fissa.
- Il gestore dati NameValue utilizza il nome ed il valore degli attributi dell'oggetto di business.

Un gestore dati con metadati gestisce ogni oggetto di business object che che può supportare in base ai metadati*codificati* nella definizione di oggetto di business piuttosto che alle informazioni codificate nel gestore dati. Perciò, un gestore dati può gestire oggetti di business nuovi o modificati senza richiedere modifiche al codice del gestore dati.

Capitolo 2. Installazione e configurazione del Gestore Dati

Questo capitolo contiene informazioni sull'installazione e configurazione del gestore dati. Descrive inoltre come configurare un connettore per supportare i gestori dati.

Questo capitolo contiene le sezioni seguenti:

Installazione Automatica dei gestori dati.

Con l'installazione di WebSphere Business Integration Server Express, i gestori dati vengono installati automaticamente. I gestori dati XML, EDI, e Complex Data vengono installati con Windows. I gestori dati XML e EDI sono installati con Linux e i5/OS e i5/OS.

Una volta terminata l'installazione di WebSphere Business Integration Server Express, i file in Tabella 6vengono automaticamente installati nella directory del prodotto nel proprio sistema.

Tabella 6. Struttura dei file installati per i gestori dati in WebSphere Business Integration Server Express

Directory	Descrizione
`DataHandlers`	Contiene i seguenti file: • `CwXMLDataHandler.jar` , contenente il gestore dati XML separato • `CwDataHandler.jar` per versioni compilate di gestori dati IBM • un file jar vuoto, `CustDataHandler.jar`, previsto per contenere tutti i gestori dati personali che potrebbero essere sviluppati.
`repository\sadk`	Contiene file di testo con definizioni meta oggetto per gestori dati richiamati nel contesto dell'interfaccia di Server Access.
`DevelopmentKits\edk\DataHandler`	Contiene un file maschera (`StubDataHandler.java`) per un gestore dati personalizzato e un file batch (`StubDataHandler.java`) per compilare un gestore dati personalizzato.
`DevelopmentKits\edk\DataHandler\Samples`	Contiene un codice sorgente per l'esempio FixedWidth, NameValue e Gestori dati delimitati.
`repository\DataHandlers`	Contiene file di testo con definizioni meta oggetto per gestori dati richiamati nel contesto di un connettore.

Configurazione dei gestori dati

I meta oggetti del gestore dati consentono ad un connettore o al processo Server Access Interface di creare un'istanza per un gestore dati basato sul MIME type di un file di immissione o il MIME type specificato in una richiesta oggetto di business. Per configurare un gestore dati, assicurarsi che i suoi meta oggetti siano correttamente inizializzati e disponibili per i componenti richiedenti (un connettore o un client di accesso).

Nota: Ciascun gestore dati IBM utilizza proprietà di configurazione che sono definite nei meta oggetti del gestore dati. Tuttavia, un gestore dati personalizzato può o non può utilizzare meta oggetti per le sue proprietà di configurazione. Per ulteriori informazioni, consultare "Utilizzo dei meta-oggetti del gestore dati" a pagina 179.

Nel supporto dei gestori dati IBM, IBM distribuisce i meta oggetti del gestore dati elencati in Tabella 7.

Tabella 7. meta oggetti del gestore dati IBM

Livello meta oggetto	Quantità	Ubicazione
Livello superiore		
Per InterChange Server Express	Uno	`repository\edk`
Per il connettore	Uno	`repository\DataHandlers`
Elemento secondario	Uno per ciascuno gestore dati	`repository\DataHandlers`

Per configurare l'utilizzo di uno o più gestori dati per l'utilizzo da parte di un richiedente, è necessario:

- In un meta oggetto di livello superiore, fornire al richiedente i tipi MIME supportati e i gestori dati ad essi associati.
- In un meta oggetto secondario, fornire al richiedente le informazioni di configurazione appropriate per il funzionamento desiderato del gestore dati.

meta oggetti principali

Un meta oggetto di livello superiore del gestore dati associa un MIME type con un meta oggetto secondario del gestore dati. Il meta oggetto secondario fornisce informazioni di configurazione, che includono sempre il nome della classe gestore dati per creare un'istanza. Perciò, un meta oggetto di livello superiore associa un MIME type con un gestore dati. Tutti i componenti richiedenti con accesso ad un particolare meta oggetto possono richiamare ciascuno dei gestori dati il cui MIME type appare in tale meta oggetto.

E' possibile controllare quale gestore dati un componente richiedente può supportare raggruppando gli attributi MIME type appropriati in un singolo meta oggetto di livello superiore e avendo fornito ai componenti richiedenti il nome del meta oggetto contenente i gestori dati richiesti per l'utilizzo. IBM fornisce i seguenti meta oggetti principali del gestore dati:

- `MO_Server_DataHandler` per identificare i gestori dati disponibili per i client di accesso che richiamano i gestori dati.
- `MO_DataHandler_Default` per identificare i gestori dati disponibili per i connettori che richiamano i gestori dati.

meta oggetto MO_Server_DataHandler

Il Server Access Interface utilizza il meta oggetto `MO_Server_DataHandler` per identificare i gestori dati utilizzabili. La versione fornita di `MO_Server_DataHandler` non è configurata per supportare nessun MIME type. Include unicamente un singolo attributo fittizio. E' possibile personalizzare tale meta oggetto per supportare qualsiasi gestore dati installato con InterChange Server Express. Se si desidera che i propri client di accesso supportino un MIME type, rinominare l'attributo `dummy` nel meta oggetto di livello superiore `MO_Server_DataHandler` con il nome del MIME type supportato e fornire il meta oggetto secondario associato a tale MIME type.

Ad esempio, per fornire client di accesso con supporto per il MIME type `text_xml`, rinominare l'attributo `dummy` in `text_xml` e fornire il nome del meta oggetto secondario associato al MIME type così come il tipo attributo. Tale meta oggetto secondario configura il gestore dati XML. Figura 9 illustra un meta oggetto `MO_Server_DataHandler` contenente un attributo, `text_xml`, il quale rappresenta il meta oggetto secondario `MO_DataHandler_DefaultXMLConfig` .

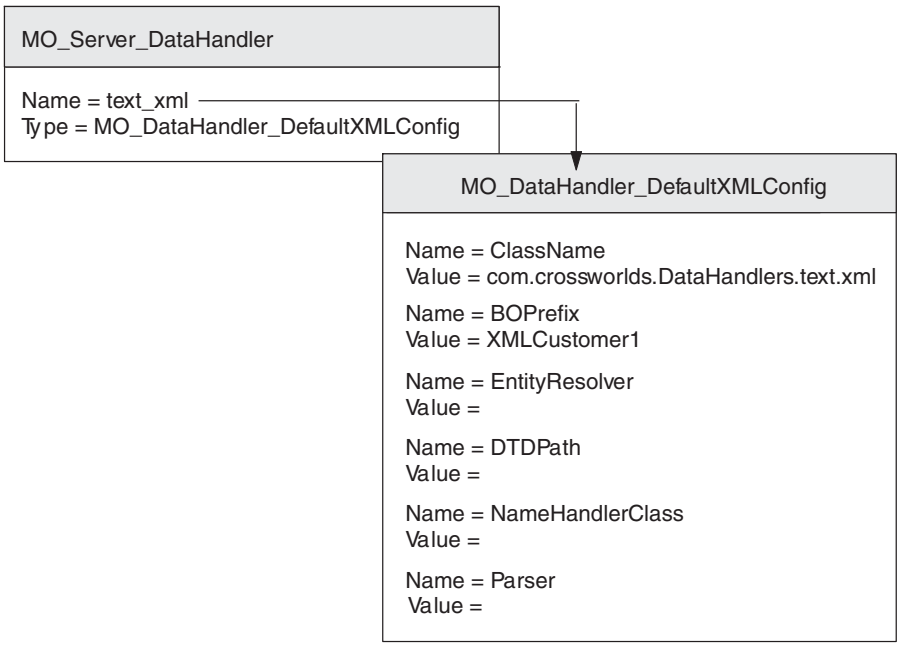

Figura 9. meta oggetto MO_Server_DataHandler

Se si desidera che i propri client di accesso supportino tipi MIME aggiuntivi, è necessario definire un nuovo attributo nel meta oggetto di livello superiore `MO_Server_DataHandler` per ciascun MIME type e fornire il meta oggetto secondario associato per tale tipo di MIME. Se si richiama più di un gestore dati, è necessario definire un meta oggetto secondario per ciascuna istanza del gestore dati. Per fornire supporto ad un MIME type aggiuntivo, è possibile sia:

- Aggiungere ognuno dei tipi MIME supportati dai gestori dati IBM (fare riferimento a Tabella 3 a pagina 5 e Tabella 4 a pagina 5) al meta oggetto di livello superiore del gestore dati.
- Definire il proprio MIME type personalizzato e il meta oggetto secondario, finché non si ottiene un gestore dati che lo supporti. Per ulteriori informazioni, consultare "Modifica del meta-oggeto di livello superiore" a pagina 197.

Nota: Il nome del meta oggetto di livello superiore del server per gestori dati *deve* essere il nome predefinito del `MO_Server_DataHandler`, ma è possibile configurare il meta oggetto di livello superiore affinché contenga qualsiasi numero di meta oggetti secondari.

meta oggetto MO_DataHandler_Default

Per impostazione predefinita, un il connettore utilizza il meta oggetto `MO_DataHandler_Default` per identificare i gestori dati utilizzabili. La versione fornita di `MO_DataHandler_Default` è configurata per supportare i tipi MIME di tutti i gestori dati IBM (inclusi alcuni gestori dati che non sono trattati in questo manuale).

Se si desidera che il proprio connettore supporti tipi MIME differenti, è necessario assicurarsi che esista un attributo nel meta oggetto `MO_DataHandler_Default` per ciascun MIME type che si desidera il connettore supporti. E' necessario che tale attributo specifichi il MIME type appropriato e rappresenti il meta oggetto associato per tale MIME type. Per fornire supporto ad un MIME type aggiuntivo, è possibile determinare il proprio MIME type personalizzato e il meta oggetto secondario, finchè non si ottiene un gestore dati che lo supporti. Per ulteriori informazioni, consultare "Modifica del meta-oggeto di livello superiore" a pagina 197.

Nota: E' possibile modificare il nome del meta oggetto di livello superiore per connettori per farli corrispondere ad un preciso connettore o ad un preciso oggetto di business o anche ad un preciso tipo di file che il connettore richiede per l'elaborazione. L'utilizzo di qualsiasi oggetto, comunque, deve essere supportato dalla definizione del connettore, in modo che se si utilizza un differente oggetto di livello superiore è necessario configurare la definizione del connettore per supportarlo. Per ulteriori informazioni, consultare "Configurazione dei connettori per l'utilizzo dei gestori dati" a pagina 26.

Figura 10 mostra un meta oggetto di livello superiore del gestore dati per connettori il quale definisce due gestori dati: XML e NameValue.

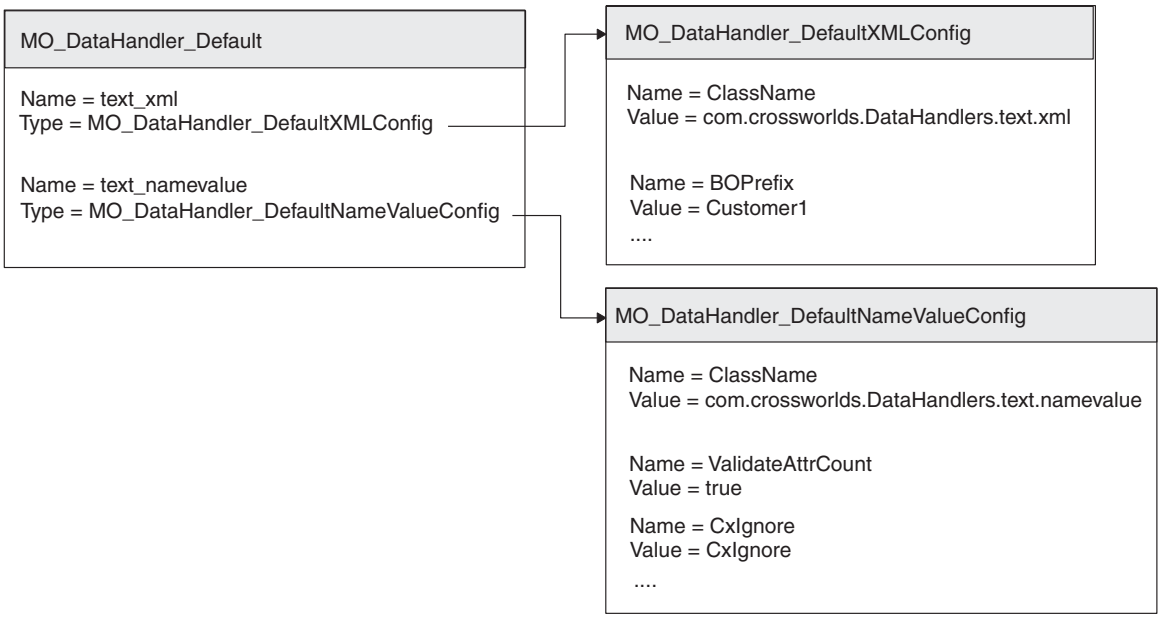

Figura 10. Meta oggetto di esempio per due differenti gestori dati

Nota: Perché un connettore sia in grado di accedere ad un gestore dati, il meta oggetto di livello superiore del gestore dati deve essere presente nell'elenco degli oggetti supportati dal connettore. Altrimenti, il connettore non può caricare il meta oggetto all'avvio.

Meta oggetti secondari

Un meta oggetto secondario del gestore dati è un oggetto di business semplice contenente informazioni di configurazione per inizializzare un gestore dati. Differenti tipi di gestori dati hanno differenti requisiti di configurazione, allo stesso modo i meta oggetti secondari hanno differenti attributi. Tali informazioni di

configurazione personalizzano il funzionamento dell'istanza gestore dati. Perciò, un insieme di valori attributo in un meta oggetto secondario definisce una configurazione specifica, che alternativamente viene associata con un singolo funzionamento del gestore dati. Ciascun richiedente che accede ad un singolo meta oggetto secondario richiede il funzionamento del gestore dati associato che le informazioni di configurazione definiscono.

- Se tutti i richiedenti che accedono ad un dato meta oggetto di livello superiore richiedono *solo uno* stato di un singolo gestore dati, forniscono appropriate informazioni di configurazione in un meta oggetto secondario e lo associano con tipi MIME del gestore dati nel meta oggetto di livello superiore.

  Ad esempio:

  – Assicurarsi che, per tutti i connettori che accedono al singolo funzionamento del gestore dati il meta oggetto secondario sia associato con i tipi MIME del gestore dati nel meta oggetto di livello superiore per connettori (`MO_DataHandler_Default` per impostazione predefinita).
  – Assicurarsi che, per tutti i client di accesso che accedono al singolo funzionamento del gestore dati il meta oggetto sia associato con i tipi MIME del gestore dati nel meta oggetto di livello superiore per il server, `MO_Server_DataHandler`.

- Se i richiedenti che accedono ad un dato meta oggetto di livello superiore richiedono *più di un solo* funzionamento di un singolo gestore dati, creare un meta oggetto secondario con le appropriate informazioni di configurazione per ciascun funzionamento del gestore dati e associare ciascun meta oggetto secondario con un nome MIME type univoco (nel meta oggetto di livello superiore del gestore dati).

  IBM raccomanda di denominare il meta oggetto secondario con una combinazione univoca MIME type/tipo secondario:

  `testo_`*`MIMEtype_sottotipo`*

  dove:

  – Il MIME type (*`MIMEtype`*) rappresenta il MIME type che il gestore dati supporta.
  – Il tipo secondario MIME (*`sottotipo`*) rappresenta il singolo funzionamento del gestore dati.

  Ad esempio, se tutti i connettori possono supportare *sia* il gestore dati XML predefinito che una versione SGML, è possibile creare i seguenti tipi MIME: `text_xml` and `text_xml_sgml`. Tenere presente le i nomi MIME type possono contenere solo caratteri alfanumerici e i caratteri speciali punto (.) e di sottolineatura (_).

IBM fornisce un meta oggetto secondario del gestore dati per ciascuno dei gestori dati forniti, come Tabella 8 mostra.

Tabella 8. Meta oggetti secondari del gestore dati

meta oggetto secondario	**Per ulteriori informazioni**
`MO_DataHandler_DefaultXMLConfig`	"Configurazione del gestore dati XML" a pagina 38
`MO_DataHandler_DefaultEDIConfig`	"Configurazione del gestore dati EDI" a pagina 94
`MO_DataHandler_DefaultFixedWidthConfig`	"Configurazione del gestore dati FixedWidth" a pagina 140
`MO_DataHandler_DefaultDelimitedConfig`	"Configurazione del gestore dati delimitato" a pagina 148
`MO_DataHandler_DefaultNameValueConfig`	"Configurazione del gestore dati NameValue" a pagina 156

Tabella 8. Meta oggetti secondari del gestore dati (Continua)

meta oggetto secondario	Per ulteriori informazioni
`MO_DataHandler_DefaultRequestResponseConfig`	"Configurazione del gestore dati di richiesta-risposta" a pagina 132
`MO_DataHandler_Complex`	"Configurazione del gestore dati Complex Data" a pagina 167

Configurazione dei connettori per l'utilizzo dei gestori dati

Se un gestore dati è in esecuzione nel contesto di un connettore, è necessario configurare il connettore per utilizzare un gestore dati:

- E' necessario che un connettore debba avere accesso al meta oggetto del gestore dati in modo da creare un'istanza per il gestore dati.

  Prima di richiamare un gestore dati per la prima volta, un connettore imposta una proprietà statica nella classe base del gestore dati per il nome del meta oggetto di livello superiore del gestore dati. Da questo meta oggetto di livello superiore, il gestore dati ottiene le sue informazioni di configurazione. Ogni volta che il gestore dati viene istanziato successivamente, si ottengono le proprietà di configurazione per tale istanza del gestore. E' necessario che il gestore dati abbia accesso a tali informazioni di configurazione per poter funzionare.
- Per istanziare il gestore dati, il connettore deve conoscere o il nome per la classe del gestore dati o il MIME type dei dati.

  Quando un connettore richiede `createHandler()` per richiamare un gestore dati, viene passato o attraverso il nome classe o il MIME type per i dati.
  - Se il connettore viene passato attraverso il MIME type, il metodo `createHandler()` verifica il meta oggetto di livello superiore del gestore dati per un attributo il cui nome corrisponda al MIME type. Se viene trovato un attributo corrispondente, il metodo `createHandler()` verifica il valore delle l'attributo `ClassName` nel meta oggetto secondario che è associato con il MIME type.
  - Se il connettore viene passato attraverso un nome classe, il metodo `createHandler()` crea un'istanza del gestore dati di questo nome classe.

  Se il connettore non viene passato attraverso il nome classe corretto o MIME type, il processo di creazione di un'istanza non riesce. Per ulteriori informazioni, consultare "Identificazione della classe del gestore dati" a pagina 13.

I connettori sono configurati per ottenere tali informazioni di configurazione in differenti modi. Ad esempio:

- L'adattatore per XML possiede la proprietà di configurazione, `DataHandlerConfigMO`, che specifica il nome del meta oggetto di livello superiore. Se tale proprietà non è compilata, il connettore non può trovare il meta oggetto. Inoltre, qualsiasi oggetto di business di livello superiore per il connettore XML deve avere un attributo `MIMEtype` che specifichi il MIME type dei dati nell'oggetto di business. Il connettore utilizza il valore dell'attributo `MimeType` per richiamare il gestore dati appropriato.
- L'adattatore per JText ha il suo meta oggetto di configurazione, che contiene gli attributi `ClassName`, `DataHandlerConfigMO` e `MimeType` per specificare rispettivamente il nome della classe, il gestore dati il meta oggetto, e il MIME type per un file.

Altri connettori possono avere differenti modi di configurare l'utilizzo di un gestore dati. Per ulteriori informazioni fare riferimento alla guida adattatore per un connettore.

Se un connettore non può trovare il meta oggetto di livello superiore del gestore dati, o non può determinare il nome classe o il MIME type, allora non può creare il gestore dati. Perciò, quando si configura un connettore per utilizzare un gestore dati, assicurarsi di:

1. Determinare come configurare il nome del meta oggetto di livello superiore del gestore dati per il connettore. Assicurarsi che l'ortografia del nome meta oggetto sia corretta.
2. Determinare come configurare il MIME type. Assicurarsi che il MIME type sia scritto correttamente.
3. Assicurarsi che il meta oggetto di livello superiore del gestore dati sia nell'elenco degli oggetti supportati per il connettore.
4. Assicurarsi che il meta oggetto secondario per il gestore dati abbia il valore del nome classe del gestore dati (nell'attributo `ClassName`) correttamente specificato.

Capitolo 3. Gestore dati XML

Il Gestore dati per XML IBM WebSphere Business Integration Server Express, chiamato il *gestore dati XML*, converte oggetti di business in documenti XML e i documenti XML in oggetti di business.

Nota: Il gestore dati XML supporta XML versione 1.0.

Tale capitolo descrive come il gestore dati XML elabora i documenti XML e come definire gli oggetti di business che devono essere elaborati dal gestore dati XML. Descrive inoltre come configurare il gestore dati XML. Questo capitolo contiene le sezioni seguenti:

Panoramica

Il gestore dati XML è un modulo di conversione dati il cui ruolo principale è convertire gli oggetti di business in documenti XML e viceversa. Un documento XML è costituito da dati serializzati con `testo/xml` MIME type. Il gestore dati XML può essere utilizzato dai connettori e dai client di accesso.

Questa panoramica fornisce le seguenti informazioni sul gestore dati XML:

Elaborazione documenti XML e oggetti di business

I documenti XML utilizzano una maschera, chiamata schema, per definire la loro struttura. Tabella 9 mostra i più comuni modelli di dati per la definizione di tale schema.

Tabella 9. Modelli di dati XML

Modello di dati XML	Per ulteriori informazioni
Definizioni del tipo di documento (DTD)	"Documenti XML che utilizzano DTD" a pagina 42
Documenti dello schema	"Documenti XML che utilizzano documenti dello schema" a pagina 55

Così come un DTD o un documento dello schema descrivono la struttura di un documento XML, allo stesso modo la definizione di un oggetto di business descrive la struttura di un oggetto di business. Il gestore dati XML utilizza le

definizioni dell'oggetto di business quando converte tra oggetti di business e documenti XML. Determina,inoltre, come effettuare la conversione utilizzando la struttura della definizione dell'oggetto di business e le sue informazioni di applicazione. Una definizione di oggetto di business costruita correttamente assicura che il gestore dati può convertire correttamente un oggetto di business in un documento XML e viceversa. Prima che il gestore dati XML possa effettuare una conversione tra un documento XML e un oggetto di business, deve essere in grado di localizzare la definizione di oggetto di business associato.

L'utilizzo del gestore dati XML per convertire un documento XML in un oggetto di business o viceversa richiede che si verifichino i seguenti passaggi.

Tabella 10. Utilizzo del gestore dati XML

Passo	Per ulteriori informazioni
1. Le definizioni dell'oggetto di business che descrivono la struttura XML e la struttura dell'oggetto di business devono esistere ed essere disponibili per il gestore dati XML quando li esegue.	"Requisiti per le definizioni oggetto di business" a pagina 33 "Creazione di definizioni di un oggetto di business dai DTD" a pagina 54
2. Il gestore dati XML deve essere configurato per il proprio ambiente.	"Configurazione del gestore dati XML" a pagina 38
3. Il gestore dati XML deve essere richiesto da un connettore (o client di accesso) per eseguire l'appropriata operazione dati:	
a) Operazione dati: riceve un oggetto di business dal richiedente, converte l'oggetto di business in un documento XML e invia il documento XML al richiedente.	"Conversione degli oggetti di business in documenti XML" a pagina 85
b) Operazione dati: Riceve un documento XML dal richiedente e utilizza il gestore nome e parser SAX per creare un oggetto di business. Quindi restituisce l'oggetto di business al richiedente.	"Conversione di documenti XML in oggetti di business" a pagina 88

componenti del gestore dati XML

Il gestore dati XML utilizza i seguenti componenti per convertire dati XML in oggetti di business:

- Gestore nome
- API semplice per XML parser (SAX)
- (Facoltativo) Se il documento XML possiede un DTD che include riferimenti di un'entità, il gestore dati XML utilizza un componente addizionale—l'applicazione che risolve entità—per risolvere i riferimenti.

Figura 11 illustra i componenti del gestore dati XML e le relazioni che intercorrono tra di loro. Questi componenti verranno descritti nelle seguenti sezioni.

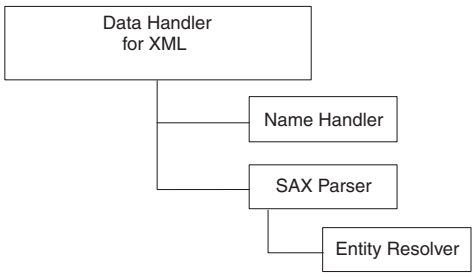

Figura 11. componenti del gestore dati XML

Gestore nome

Il gestore dati XML utilizza il gestore nome per estrarre il nome del oggetto di business da un messaggio XML. Il gestore dati invoca un'istanza del gestore nome in base al valore dell'attributo `NameHandlerClass` nel meta oggetto secondario del gestore dati XML:

- Se un nome classe viene fornito nell'attributo `NameHandlerClass`, il gestore dati XML utilizza tale gestore nome per determinare il nome dell'oggetto di business.
- Se non viene fornito alcun nome classe, il gestore dati utilizza il gestore nome predefinito per determinare il nome dell'oggetto di business. Il gestore nome predefinito utilizza il nome del elemento root nel documento XML e `BOPrefix` per comporre il nome dell'oggetto di business:

  `BOPrefix_rootElement`

Per ulteriori informazioni su come creare un gestore nome personalizzato, fare riferimento a "Creazione di un gestore nome personalizzato XML" a pagina 89.

parser SAX

Se un parser *non* è specificato nella proprietà Valore predefinito della `attributo Parser` nel meta oggetto secondario XML, il gestore dati utilizza il Parser SAX predefinito:

```
org.apache.xerces.parsers.SAXParser
```

Per utilizzare un parser di convalida, è possibile eseguire uno dei seguenti passaggi:

- Impostare la proprietà del Valore predefinito della `attributo Validation` nel meta oggetto secondario XML su true. Il Valore predefinito che IBM fornisce per tale attributo è false.
- Per utilizzare un parser SAX di convalida IBM, modificare il nome classe nella proprietà del Valore predefinito dell'attributo `Parser` in:

  ```
  com.ibm.xml.parsers.ValidatingSaxParser
  ```

  Se le proprie definizioni oggetto di business sono basate sui DTD, utilizzare l'applicazione locale che risolve entità e fornire un Valore predefinito per il DTD (document type definition) o percorso schema nell'attributo `DTDPath`. Assicurarsi di posizionare tutti i file DTD o Schema nella posizione specificata nell'attributo DTDPath.

Nota: Quando si utilizza un parser di convalida, assicurarsi che si utilizzi la corretta `EntityResolver` e che il `DTDPath` sia impostato correttamente. Per ulteriori informazioni su come effettuare quest'operazione, fare riferimento a "Configurazione del gestore dati XML" a pagina 38.

In alternativa, è possibile utilizzare Parser SAX non di verifica di IBM. Per utilizzare tale programma di verifica, impostare la proprietà del Valore predefinito dell'attributo Parser del meta oggetto secondario XML al valore `com.ibm.xml.parsers.SAXParser`.

Applicazione che risolve entità

L'applicazione che risolve entità specifica come il parser SAX risolve riferimenti esterni (come referenzi per DTD e documenti dello schema) nei dati XML. Se il documento XML contiene riferimenti di una entità, il parser SAX richiama un'istanza dell'applicazione che risolve entità utilizzando la `attributo EntityResolver` nel meta oggetto di configurazione XML del gestore dati.

I riferimenti esterni sono gestiti in maniera differente in base alla classe dell'applicazione che risolve entità che `EntityResolver` specifica. Tabella 11 mostra le classi dell'applicazione che risolve entità che il gestore dati XML fornisce.

Tabella 11. Classi applicazione che risolve entità per il gestore dati XML

Classe applicazione che risolve entità	Descrizione
`DefaultEntityResolver`	Questa classe è l'applicazione che risolve entità predefinita. Se tale applicazione che risolve entità viene richiamata, tutti i riferimenti esterni sono ignorate.
`LocalEntityResolver`	L'applicazione locale che risolve entità elabora i riferimenti esterni come nomi file locali. Il suo comportamento dipende dal modello di dati utilizzato per la sua verifica: • Se si utilizzano DTD per la verifica, l'applicazione locale che risolve entità sostituisce il percorso in `systemID` col valore della `DTDPath` attributo meta oggetto, se `systemID` inizia con `file://` o `http://` e l'attributo `DTDPath` è impostato. Il riferimento esterno viene ignorato se `systemID` non è un nome percorso o l'attributo `DTDPath` non è impostato. • Se i documenti dello schema vengono utilizzati per la verifica, l'applicazione locale che risolve entità sostituisce il percorso che il attributo`schemaLocation` o `noNamespaceSchemaLocation` specifica con il valore dell'attributo meta oggetto `DTDPath`, se il percorso inizia con `file://` o `http://`, oppure contiene un nome file DOS (ad esempio, `"D:\xmlschemas\test"`).
`URIEntityResolver`	Tale applicazione che risolve entità elabora riferimenti esterni come nomi file locali o URL scaricabili. Risolve dinamicamente il riferimento esterno in entrambi questi casi: • Se vengono utilizzati DTD per la verifica: se `DOCTYPE` contiene un valore `SYSTEM` che inizia con `http://` o un file:// • Se i documenti dello schema vengono utilizzati per la verifica: se l'attributo `schemaLocation` o `noNamespaceSchemaLocation` inizia con `http:// o con file://` L'applicazione che risolve entità quindi apre una connessione HTTP e scarica il DTD o il documento dello schema dal sito Web specificato. **Attenzione:** Il gestore dati XML *non* memorizza nella cache i DTD o i documenti dello schema. Quando il gestore dati utilizza la classe `URIEntityResolver`come sua applicazione che risolve entità, apre una connessione HTTP *ogni volta che* analizza il documento XML. Perciò, il traffico di rete può influenzare la prestazione del gestore dati XML.

Nota: E' necessario che tutte le classi applicazione che risolve entità in Tabella 11 abbiano il seguente prefisso di classe:

com.crossworlds.DataHandlers.xml

Se i propri documenti XML utilizzano documenti dello schema, qualsiasi schema esterno che il documento dello schema include è elaborato come entità esterna. Perciò, il parser SAX richiama un'applicazione che risolve entità per risolvere i documenti dello schema inclusi. Se il documento XML utilizza `schemaLocation` o `noNamespaceSchemaLocation` per specificare le posizioni schema, è possibile impostare l'attributo `EntityResolver` sia per `LocalEntityResolver` che per `URIEntityResolver` per l averifica dei documenti dello schema esterni (sia inclusi che importati).

Se è necessario specificare un altro modo per trovare entità esterne, è necessario creare un'applicazione che risolve entità personalizzata. Per ulteriori informazioni sulla creazione di un'applicazione che risolve entità personalizzata, fare riferimento a "Creazione di un'applicazione che risolve entità personalizzata" a pagina 91.

Requisiti per le definizioni oggetto di business

Per assicurarsi che le definizioni oggetto di business corrispondano ai requisiti del gestore dati XML, utilizzare le linee guida in questa sezione, che include:

- "Struttura oggetto di business"
- "Proprietà dell'attributo oggetto di business" a pagina 34
- "informazioni specifiche sull'applicazione" a pagina 37
- "Verbi dell'oggetto di business" a pagina 38

Una definizione dell'oggetto di business costruita correttamente assicura che il gestore dati può convertire correttamente un oggetto di business in un documento XML e viceversa. Per ulteriori informazioni su come creare oggetti di business per il gestore dati XML, fare riferimento a "Creazione di definizioni di un oggetto di business dai DTD" a pagina 54.

Struttura oggetto di business

Per rappresentare un DTD o documento dello schema richiede almeno due definizioni oggetto di business:

- Il oggetto di business*di livello superiore* rappresenta le informazioni che definiscono un DTD o un documento dello schema ed è necessario che contenga i seguenti attributi:
  - Un attributo chiamato `XMLDeclaration` per rappresentare la versione XML

    Questo attributo deve avere la tag `type=pi` nelle relative informazioni specifiche sull'applicazione.
  - Un attributo per rappresentare l'elemento root in DTD o documento dello schema

    E'necessario che questo attributo abbia come tipo relativo un oggetto di business a cardinalità singola, il cui tipo è la definizione dell'oggetto di business per l'elemento root del DTD o documento dello schema. L'ODA XML ottiene il nome di questo elemento root dalla `Root` ODA proprietà di configurazione. E' necessario che le informazioni specifiche sull'applicazione elenchino il nome di tale elemento con la tag `elem_name`.

    Nota: La tag `elem_name` sostituisce la sintassi precedente, che richiede solo il nome dell'elemento XML nelle informazioni specifiche sull'applicazione attributo oggetto di business. Il gestore dati XML supporta ancora la vecchia sintassi per la compatibilità delle versioni precedenti delle

definizioni degli oggetti di business esistenti. Tuttavia, l'ODA XML utilizza la nuova sintassi quando genera le definizioni dell'oggetto di business.

- Una definizione di oggetto di business *elemento root* rappresentano l'elemento root del documento definizione XML. E' possibile indicare a ODA XML ODA quale elemento deve essere considerato l'elemento root attraverso la sua proprietà di configurazione `Root`. Tale elemento contiene un attributo per ciascuno dei componenti XML nell'elemento root.

Un oggetto di business che viene elaborato dal gestore dati XML utilizzando le definizioni dell'oggetto di business dai DTD o dalle definizioni schema deve obbedire alle seguenti regole:

- E' necessario che ogni tag nel documento XML abbia un attributo associato nell'oggetto di business. La definizione dell'oggetto di business fornisce il tipo di attributo di oggetto di business e vengono memorizzate le informazioni specifiche sull'applicazione per tale attributo. Tali informazioni vengono determinate dalla struttura e dal contenuto dell'elemento XML.
- Nella definizione di oggetto di business per un elemento XML, tutti gli attributi che rappresentano il attributi XML devono presentare *prima* altri attributi. Il gestore dati XML presume che tali attributi per un dato elemento XML sono i *primi* attributi nelle definizione di oggetto di business.

Nota: E' necessario che un oggetto di business contenga abbastanza dati in modo che il gestore dati XML possa creare un documento XML valido. Evitare di mandare oggetti di business del gestore dati senza dati.

Questo manuale fornisce le seguenti informazioni sulla struttura delle definizioni di oggetto di business per DTD e documenti dello schema:

Modello di dati	Per ulteriori informazioni
DTD (document type definition)	"Struttura dell'oggetto di business per DTD" a pagina 42
documento dello schema	"Struttura di un oggetto di business per documenti dello schema" a pagina 56

Proprietà dell'attributo oggetto di business

Le definizioni dell'oggetto di business definiscono gli attributi. Ogni attributo ha diverse proprietà che forniscono informazioni sull'attributo. Questa sezione descrive come il gestore dati XML interpreta numerose di queste proprietà e descrive come impostarle quando si modifica una definizione di oggetto di business.

Proprietà dell'attributo Name

E' necessario che ogni attributo dell'oggetto di business abbia una denominazione univoca. Il nome Elemento XML o Attributo XML sono sempre specificati nelle tag elem_name o attr_name. In questo caso, il nome dell'elemento (o attributo) XML specificato nella tag elem_name (o attr_name) delle informazioni specifiche sull'applicazione dell'attributo contiene caratteri speciali. Tuttavia, il nome dell'attributo dell'oggetto di business (che non consente l'utilizzo di caratteri speciali) li tralascia.

Proprietà dell'attributo Type

E' necessario che ogni attributo dell'oggetto di business abbia un tipo, come un Intero, una Stringa o il tipo di un oggetto di business secondario contenuto, come di seguito:

- Per un DTD: gli elementi XML che contengono o elementi secondari o uno o più attributi non FISSI sono elaborati come oggetti di business. Gli elementi che hanno solo un valore PCDATA sono elaborati come attributi se l'elemento XML è incluso nel relativo elemento principale attraverso la cardinalità singola. Se è incluso mediante una cardinalità multipla, è rappresentato come un oggetto di business poiché le definizioni di oggetto di business non supportano valori scalari a cardinalità multipla (ad esempio, un array di valori stringa).
- Per un documento dello schema: E' necessario che ogni attributo dell'oggetto di business abbia un tipo o di una stringa o il tipo di un oggetto di business secondario contenuto. Gli elementi XML che contengono o gli elementi secondari o tipi complessi sono elaborati come oggetti di business. Gli elementi XML con solo un valore di tipo semplice sono elaborati come attributi dell'oggetto di business se l'elemento XML è incluso nel relativo elemento principale mediante la cardinalità singola. Se è incluso mediante una cardinalità multipla, è rappresentato come un oggetto di business poiché le definizioni di oggetto di business non supportano valori scalari a cardinalità multipla (ad esempio, un array di valori stringa).

Nota: Tutti gli attributi semplici dovrebbero essere del tipo `String`

Proprietà degli attributi Chiave e Chiave esterna

E' necessario che ogni oggetto di business abbia almeno un attributo della chiave primaria, che è specificato impostando per un attributo la proprietà Key su true. L'impostazione della proprietà Foreign Key è facoltativa e dipende dalla struttura del documento XML. Questa sezione fornisce le seguenti informazioni riguardo alle proprietà degli attributi Key e Foreign Key:

- "Designare la chiave nella definizione dell'oggetto di business"
- "Chiavi di gestione e obbligatorietà" a pagina 36

Designare la chiave nella definizione dell'oggetto di business: Nelle precedenti versioni degli strumenti di creazione della definizione oggetto di business XML (come ad esempio XMLBorgen, Edifecs SpecBuilder e ODA XML), lo strumento di creazione designava l'attributo ObjectEventId come la chiave di un oggetto di business XML principale. Tuttavia, a partire da questa versione, Business Object Designer Express non consente più di salvare una definizione di oggetto di business che abbia l'attributo ObjectEventId specificato come chiave.

A causa di tale restrizione, la versione corrente dell'ODA XML effettua ora le seguenti operazioni:

- In ogni oggetto di business secondario, imposta il primo attributo come chiave.
- Nell'oggetto di business principale, *non* imposta un attributo chiave.

Per fornire una chiave ad una definizione di oggetto di business principale che l'ODA XML genera, è necessario portare la definizione di oggetto di business in Business Object Designer Express e analizzare la propria definizione di oggetto di business per determinare l'attributo appropriato da designare come chiave. E' necessario modificare l'attributo chiave della definizione dell'oggetto business *prima* di salvare la definizione dell'oggetto di business in Business Object Designer Express.

Nota: l'L'ODA XML sostituisce in anticipo gli strumenti di creazione di definizione dell'oggetto di business XML (come ad esempio XMLBorgen e Edifecs SpecBuilder). Tuttavia, solo l'ODA XML esegue questi passaggi speciali per evitare l'assegnazione dell'attributo ObjectEventId come attributo della chiave dell'oggetto di business principale. Se si hanno le definizioni di oggetti di business XML esistenti che sono stati generati con qualsiasi precedente strumento di creazione di definizione dell'oggetto di business XML (inclusa una versione precedente dell'ODA XML), tali definizioni dell'oggetto di business potrebbero ancora utilizzare ObjectEventId come una chiave. Sarebbe necessario analizzare queste definizioni di oggetto di business se si trasferiscono i relativi oggetti di business alla versione corrente. Tralasciare di impostare un attributo chiave appropriato nella propria definizione dell'oggetto di business può avere un influenza negativa sulla esecuzione dell'evento della funzione di sequenza dell'evento.

Chiavi di gestione e obbligatorietà: Questo manuale fornisce le seguenti informazioni riguardo alla relazione tra le chiavi e l'obbligatorietà:

Modello di dati	Per ulteriori informazioni
DTD (document type definition)	"Proprietà dell'attributo oggetto di business per DTD" a pagina 44
documento dello schema	"Proprietà dell'attributo di un oggetto di business per documenti dello schema" a pagina 64

Proprietà dell'attributo Required

Se questa proprietà è specificata per un attributo contenente un oggetto di business secondario a cardinalità singola, il gestore dati XML richiede che l'oggetto di business principale contenga un oggetto di business secondario per tale attributo. Le impostazioni di Cardinalità, Key e le proprietà dell'attributo Foreign Key possono riguardare l'impostazione della proprietà attributo Required.

Questo manuale fornisce le seguenti informazioni riguardo all'obbligatorietà:

Modello di dati	Per ulteriori informazioni
DTD (Document type definition)	"Proprietà dell'attributo oggetto di business per DTD" a pagina 44
documento dello schema	"Proprietà dell'attributo di un oggetto di business per documenti dello schema" a pagina 64

Proprietà dell'attributo Cardinalità

La proprietà Cardinalità indica il numero di oggetti di business secondari consentiti in un attributo che ha una definizione di oggetto di business come il relativo tipo. L'impostazione di questa proprietà dipende dalla struttura del documento XML e dai relativi elementi. La relativa impostazione riguarda inoltre se l'attributo deve essere richiesto (la relativa proprietà Required impostata su true).

Questo manuale fornisce le seguenti informazioni riguardo alla relazione tra cardinalità e obbligatorietà:

Modello di dati	Per ulteriori informazioni
DTD (Document type definition) documento dello schema	"Proprietà dell'attributo oggetto di business per DTD" a pagina 44

Valori di attributo speciali

Un attributo di un oggetto di business ha un valore il cui tipo corrisponde alla proprietà dell'attributo Type. In aggiunta, un attributo può avere uno dei due valori speciali:

- CxIgnore

  Quando il gestore dati XML riceve un oggetto di business da InterChange Server Express, ignora *tutti* gli attributi con valore CxIgnore. E'come se tali attributi fossero invisibili per il gestore dati. Perciò, il gestore dati *non* genera un elemento XML corrispondente; ossia, *non* crea una tag XML per tale attributo (nemmeno una tag vuota). Quando il gestore dati XML riceve un file di input XML che non ha una tag XML corrispondente all'attributo dell'oggetto di business, il gestore dati assegna all'attributo un valore CxIgnore.
- CxBlank

  Quando il gestore dati XML riceve un oggetto di business da InterChange Server Express, elabora il valore dell'attributo CxBlank in base alla proprietà dell'attributo Type:
  - Per un attributo complesso (uno in cui la proprietà Type è impostata sul nome di un'altra definizione di oggetto di business), il gestore dati assume che nessun attributo complesso ha il valore CxBlank.
  - Per un attributo semplice (uno in cui la proprietà Type è impostata su un tipo di dati `stringa`), il gestore dati crea una tag vuota nel documento XML. Per i documenti XML basati sui DTD, le doppie virgolette vuote (" ") sono utilizzate come l'equivalente PCDATA di CxBlank.

Quando il gestore dati XML riceve un file di input XML che ha una tag vuota, il gestore dati assegna un valore CxBlank all'attributo dell'oggetto di business corrispondente.

informazioni specifiche sull'applicazione

Le informazioni specifiche sull'applicazione nelle definizioni di oggetto di business forniscono il gestore dati con istruzioni su come convertire gli oggetti di business in documenti XML. Le informazioni specifiche sull'applicazione consentono al gestore dati di elaborare l'oggetto di business correttamente. Perciò, se si creano nuovi oggetti di business o se ne modificano di esistenti per il gestore dati XML, assicurarsi che le informazioni specifiche sull'applicazione nella definizione dell'oggetto di business corrispondano alla sintassi prevista dal gestore dati. Il gestore dati XML può utilizzare i seguenti tipi di informazioni specifiche sull'applicazione:

- Le informazioni specifiche sull'applicazione del livello dell'oggetto di business forniscono informazioni sulla definizione dell'oggetto di business nella sua interezza.
- Le informazioni specifiche sull'applicazione del livello di attributo forniscono informazioni su un particolare attributo.

Nota: Il gestore dati XML data utilizza le informazioni specifiche sull'applicazione per far corrispondere i componenti di un documento XML con gli attributi in un oggetto di business. La lunghezza massima per le informazioni

specifiche sull'applicazione è 255 caratteri. Se il valore delle informazioni specifiche sull'applicazione è maggiore di 255 caratteri, è necessario ricostruire il proprio DTD o documento dello schema e quindi rigenerare l'oggetto di business.

Questo manuale fornisce le seguenti informazioni sulle informazioni specifiche sull'applicazione:

Modello di dati	Per ulteriori informazioni
DTD (Document type definition)	"Informazioni specifiche sull'applicazione per componenti XML nei DTD" a pagina 45
documento dello schema	"Informazioni specifiche sull'applicazione per componenti XML nei documenti dello schema" a pagina 65

Verbi dell'oggetto di business

Durante la conversione di oggetti di business in documenti XML, il gestore dati XML *non* genera XML per il verbo, né imposta un verbo durante la conversione di un documento XML in un oggetto di business. Tuttavia, le informazioni del verbo possono essere conservate in uno dei seguenti modi:

- E' possibile creare un elemento nel Un DTD o un documento dello schema per il verbo e creare un attributo dell'oggetto di business per il verbo. e' possibile quindi designare il contenuto del proprio sistema di integrazione business per copiare il verbo nell'attributo dell'oggetto di business. Il gestore dati quindi converte il verbo nell'elemento XML, così da conservare il verbo nel documento XML. Quando il documento XML viene restituito, il sistema di integrazione business può impostare il verbo secondo il valore dell'attributo dell'oggetto di business.
- E' possibile creare DTD o documenti dello schema per oggetti di business specifici e combinazioni di verbi per associare ciascuna richiesta dell'oggetto di business con il DTD o il documento dello schema per quell'oggetto di business e verbo. quando un documento XML è convertito in un oggetto di business e restituito al richiedente, il connettore può impostare il verbo che corrisponde al DTD o al documento dello schema.
- Se il connettore richiedente può essere fornito con informazioni sul verbo, può impostare il verbo nell'oggetto di business prima di inviarlo a InterChange Server Express.

Configurazione del gestore dati XML

Per configurare un gestore dati XML, è necessario assicurarsi che le relative informazioni di configurazione siano fornite nel gestore dati XML meta oggetto secondario.

Nota: Per configurare un gestore dati XML, è necessario inoltre, creare o modificare le definizioni dell'oggetto di business in modo che supportino il gestore dati. Per ulteriori informazioni, consultare "Requisiti per le definizioni oggetto di business" a pagina 33.

Per il gestore dati XML, IBM fornisce il meta oggetto secondario predefinito `MO_DataHandler_DefaultXMLConfig`. Ogni attributo in questo meta oggetto definisce una proprietà di configurazione per il gestore dati XML. Tabella 12 descrive gli attributi in questo meta oggetto secondario.

Tabella 12. Attributi del meta oggetto secondario per il gestore dati XML

Nome attributo	Descrizione	Valore predefinito fornito
`BOPrefix`	Il prefisso è utilizzato dalla classe predefinita `NameHandler` per costruire i nominativi dei nomi dell'oggetto di business. Il valore predefinito deve essere modificato per corrispondere al nome della definizione dell'oggetto di business associato. Il valore dell'attributo è sensibile al maiuscolo/minuscolo.	`XMLTEST`
`NomeClasse`	Nome della classe gestore dati da caricare per l'utilizzo con il tipo MIME specificato. Il meta oggetto di livello superiore del gestore dati ha un attributo il cui nome corrisponde al MIME type specificato e il cui tipo è il meta oggetto XML secondario (come descritto in Tabella 12).	`com.crossworlds. DataHandlers. text.xml`
`DefaultEscapeBehavior`	Se il valore di un attributo contiene caratteri speciali, richiede il gestore dati XML per effettuare l'elaborazione escape. Se le informazioni specifiche sull'applicazione dell'attributo *non* includono la tag `escape`, il gestore dati XML controlla il valore delle proprietà `DefaultEscapeBehavior` per determinare se effettuare l'elaborazione escape. Per ulteriori informazioni, consultare "Per un elemento o attributo XML che contenga caratteri speciali" a pagina 52.	`true`
`DTDPath`	Utilizzato dal gestore dati per configurare il percorso nel Definizioni del tipo documento (DTD) o schemi (XSD).	*Nessuno*
`DummyKey`	attributo Key; non utilizzato dal gestore dati ma richiesto dal sistema di integrazione business.	1
`EntityResolver`	Nome della classe da utilizzare per gestire riferimenti per entità esterne come un DTD o schema. Per ulteriori informazioni riguardo ai valori per questo attributo, fare riferimento a "Applicazione che risolve entità" a pagina 32.	*Nessuno*
`IgnoreUndefinedAttributes`	Quando questo attributo è impostato su `false`, il gestore dati XML convalida tutti gli attributi XML per le definizioni di oggetto di business e invia un'eccezione quando incontra un attributo non definito. Quando questo attributo è impostato su `true`, il gestore dati XML ignora tutti gli attributi XML non definiti, generando un messaggio di avvertimento.	`true`
`IgnoreUndefinedElements`	Quando questo attributo è impostato su `false`, il gestore dati XML convalida tutti gli elementi XML per la definizione dell'oggetto di business e invia un'eccezione quando incontra un elemento non definito nelle informazioni specifiche sull'applicazione. Quando questo attributo è impostato su `true`, il gestore dati XML ignora tutti gli elementi XML non definiti (e qualsiasi attributo incluso in questi elementi non definiti), generando un messaggio di avvertimento.	`false`
`InitialBufferSize`	Definisce la dimensione iniziale del buffer che viene utilizzato durante la conversione degli oggetti di business in XML. Imposta questo valore alla dimensione, in byte, dei propri oggetti di business XML. Impostare questo valore su un numero alto accelererà la conversione degli oggetti di business in XML serializzato.	2 MB (2,097,152 KB)

Tabella 12. Attributi del meta oggetto secondario per il gestore dati XML (Continua)

Nome attributo	Descrizione	Valore predefinito fornito
NameHandlerClass	Nome della classe da utilizzare per determinare il nome dell'oggetto di business dal contenuto di un documento XML. Modificare il valore predefinito di questo attributo se si intende creare il proprio gestore dati personalizzato. Per ulteriori informazioni, consultare "Creazione di un gestore nome personalizzato XML" a pagina 89.	com. crossworlds. DataHandlers.xml. TopElementNameHandler
Parser	Nome del pacchetto di un parser conforme a SAX per il documento XML.	*Nessuno*
UseNewLine	Impostare il valore su true se si desidera che ogni tag nel file di output XML sia su una nuova linea. (Il gestore dati XML aggiunge contenuto supplementare al documento XML sotto forma di avanzamento di linea e di ritorno a capo.) Impostare il valore su false se non si desidera alterare il file di output XML.	false
Validation	Questo valore è utilizzato dal gestore dati per specificare che viene utilizzato un parser di convalida. Il gestore dati effettua questa operazione impostando la funzione http://xml.org/sax/features/validation del parser XML4J su true. Per utilizzare un parser di convalida, il valore predefinito deve essere impostato su true. Quando Validation è impostato su true, il parser XML convaliderà il documento XML per: • Un DTD, se è presente un DTD • Un documento dello schema, se ce n'è uno specificato. In questo caso, il programma di verifica XML effettua un controllo completo schema. • DTD e documenti dello schema, se entrambi sono specificati	false
ObjectEventId	Placeholder non viene utilizzato dal gestore dati ma è richiesto dal sistema di integrazione business.	*Nessuno*

Il "Valore predefinito fornito" nella colonna Tabella 12 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente nell'oggetto di business fornito. E' necessario verificare il proprio ambiente e impostare le proprietà del Valore predefinito di tali attributi con i valori appropriati. E' necessario assicurarsi che almeno gli attributi ClassName e BOPrefix abbiano valori predefiniti.

Nota: Utilizzare Business Object Designer Express per modificare le definizioni di un oggetto di business.

Un singolo meta oggetto secondario può essere utilizzato da diverse combinazioni MIME type/sotto tipo se ciascuna delle combinazioni utilizza la stessa configurazione del gestore dati XML. Se il proprio connettore richiede differenti configurazioni del gestore dati XML per differenti MIME type, allora è necessario creare un meta oggetto secondario separato per ciascuna istanza del gestore dati. Per creare configurazioni multiple del gestore dati XML, procedere nel modo seguente:

- Copiare e rinominare il meta oggetto secondario XML predefinito. Un approccio raccomandato denominazione di un nuovo meta oggetto secondario è di fornire sotto tipi al MIME type. Ad esempio, per supportare sia il gestore dati XML

predefinito che una versione SGML, è possibile copiare il meta oggetto secondario XML predefinito e denominare questa copia `MO_DataHandler_DefaultSGMLConfig`.

- Impostare i valori predefiniti degli attributi in ogni meta oggetto secondario XML per configurare l'istanza del gestore dati.

  Creare attributi nel meta oggetto di livello superiore del gestore dati per ciascuna combinazione MIME type/subtype. Ad esempio, per supportare XML e SGML, è possibile creare i seguenti MIME type: `text_xml` e `text_xml_sgml`. Ciascuno di questi attributi dovrebbe rappresentare i relativi meta oggetti secondari associati.

  Inoltre, è possibile configurare il gestore dati XML perché supporti istanze multiple dello stesso gestore dati. In questo caso, è possibile creare un'altro attributo di livello superiore chiamato `text_xml_`*`subtype`*, dove *`subtype`* può essere il nome di un'applicazione, come in `text_xml_AppA`, un nome applicazione entità o un altro nome appropriato.

Per ulteriori informazioni su come configurare un gestore dati, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

Figura 12 mostra un esempio di un meta oggetto di livello superiore di un gestore dati e i relativi meta oggetti secondari corrispondenti. Notare che sono presenti quattro attributi nel meta oggetto di livello superiore `MO_DataHandler_XMLSample`, ma solo tre nei meta oggetti secondari. Questo perché l'attributo `Application_xml_AppC` utilizza lo stesso meta oggetto secondario dell'attributo `text_xml_AppB` per richiamare il gestore dati appropriato.

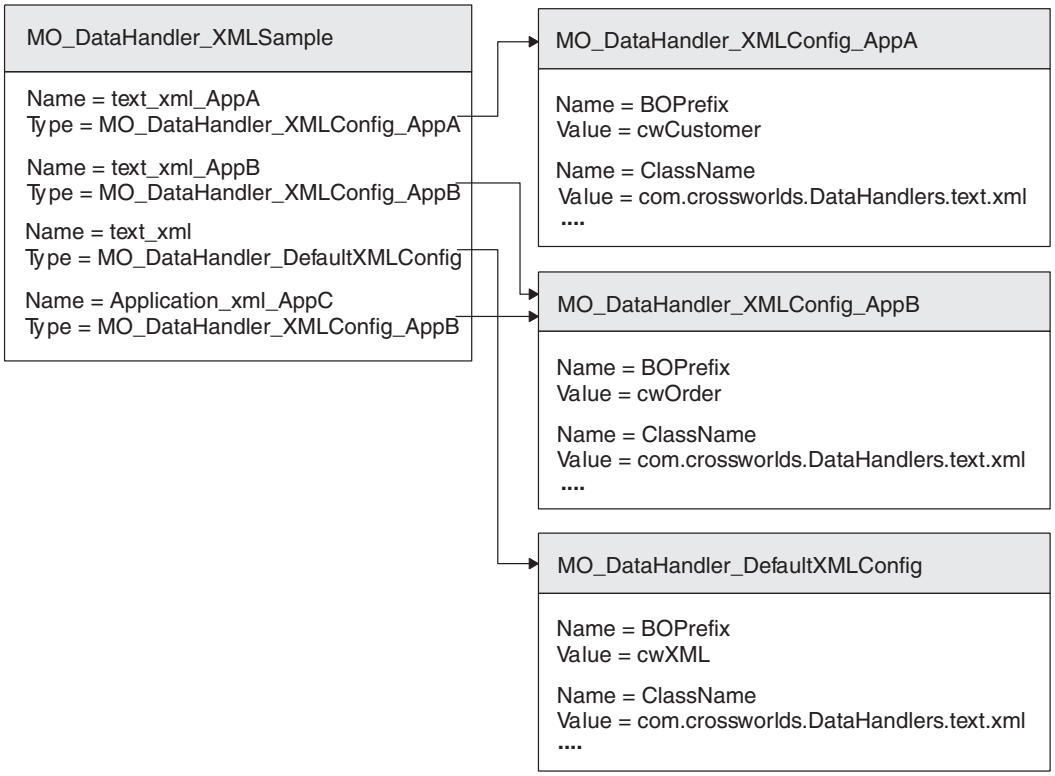

Figura 12. Esempio di meta oggetto per gestori dati XML multipli

Documenti XML che utilizzano DTD

Una *definizione del tipo di documento* (DTD) è un modello di dati per documenti XML che fornisce una sintassi speciale per descrivere la maschera del documento XML, chiamata schema. Tale DTD è un file con estensione `.dtd`. Le definizioni dell'oggetto di business che rappresentano lo schema di un documento XML utilizzano le informazioni nel DTD per conservare e registrare la struttura del documento. Questa sezione fornisce le seguenti informazioni riguardo le informazioni della struttura derivante per una definizione di un oggetto di business da un DTD:

- "Requisiti per le definizioni di un oggetto di business in base ai DTD"
- "Creazione di definizioni di un oggetto di business dai DTD" a pagina 54

Requisiti per le definizioni di un oggetto di business in base ai DTD

Per assicurarsi che le definizioni dell'oggetto di business che corrispondono ai DTD siano conformi ai requisiti del gestore dati XML, utilizzare le linee guida in questa sezione, che include:

- "Struttura dell'oggetto di business per DTD"
- "Proprietà dell'attributo oggetto di business per DTD" a pagina 44
- "Informazioni specifiche sull'applicazione per componenti XML nei DTD" a pagina 45

Una definizione oggetto di business costruita correttamente assicura che il gestore dati può convertire correttamente un oggetto di business in un documento XML e viceversa. Per ulteriori informazioni su come creare oggetti di business per il gestore dati XML, fare riferimento a "Documenti XML che utilizzano documenti dello schema" a pagina 55.

Struttura dell'oggetto di business per DTD

Per rappresentare un DTD richiede almeno le due definizioni di oggetto di business descritte in "Struttura oggetto di business" a pagina 33. Per un DTD, tali definizioni di oggetto di business hanno i seguenti requisiti aggiuntivi:

- Il oggetto di business*di livello superiore* rappresenta un DTD XML e può contenere i seguenti attributi:
  - Un attributo chiamato `DocType` per rappresentare il dichiarazione`DOCTYPE`

    Se l'ODA XML genera un attributo `DocType` nella definizione di oggetto di business di livello superiore questo dipende dalle impostazioni delle relative proprietà di configurazione`DocTypeOrSchemaLocation`. Per ulteriori informazioni, fare riferimento a "Per una dichiarazione XML DOCTYPE" a pagina 52 e "Strutture DTD supportate" a pagina 54.
  - Un attributo chiamato `XMLDeclaration` per rappresentare la versione XML

    Questo attributo deve avere la tag `type=pi` nelle relative informazioni specifiche sull'applicazione. Per ulteriori informazioni, consultare "Per le istruzioni di elaborazione XML" a pagina 78.
  - Un attributo per rappresentare l'elemento root in DTD

    Come descritto in "Struttura oggetto di business" a pagina 33, questo attributo deve avere come il relativo tipo un oggetto di business a cardinalità singola, il cui tipo sia la definizione di oggetto di business per l'elemento root. E' necessario che le informazioni specifiche sull'applicazione elenchino il

nome di tale elemento con la tag `elem_name`. Per ulteriori informazioni, consultare “Per elementi XML” a pagina 73.

- Una definizione di oggetto di business *elemento root* business rappresenta l'elemento root di DTD.

Un oggetto di business che viene elaborato dal gestore dati XML utilizzando le definizioni dell'oggetto di business in base ai DTD deve inoltre obbedire alle seguenti regole:

- E'necessario che ogni tag nel documento XML abbia un attributo associato nella definizione dell'oggetto di business. L'eccezione a questa regola è per attributi FISSI. Per definizione, gli attributi FISSI *non* sono inclusi in una definizione di oggetto di business perché tali attributi contengono dati statici. Tuttavia, se si desidera che i propri attributi FISSI siano inclusi nella definizione di oggetto di business è possibile aggiungere manualmente tali attributi alla definizione di oggetto di business.

Nota: Per un elenco dei requisiti di un oggetto di business generale, fare riferimento a “Requisiti per le definizioni oggetto di business” a pagina 33.

Un esempio per un documento XML DTD è mostrato di seguito. Il DTD viene denominato `Order` e contiene elementi che corrispondono ad una entità di applicazione `Order`.

```
<!--Order -->
<!-- Element Declarations -->
<!ELEMENT Order (Unit+)>
<!ELEMENT Unit (PartNumber?, Quantity, Price, Accessory*)>
<!ELEMENT PartNumber (#PCDATA)>
<!ELEMENT Quantity (#PCDATA)>
<!ELEMENT Price (#PCDATA)>
<!ELEMENT Accessory (Quantity, Type)>
<!ATTLIST Accessory
     Name CDATA >
<!ELEMENT Type (#PCDATA)>
```

Figura 13 mostra la struttura di un oggetto di business che potrebbe essere creato per corrispondere ad un documento XML associato con il DTD `Order`. Notare che ogni elemento XML e l'elemento attributo nel DTD `Order` hanno un attributo dell'oggetto di business corrispondente. L'oggetto di business di livello superiore contiene attributi per la dichiarazione XML, il `DOCTYPE` e l'elemento Order di livello superiore. Notare inoltre, che l'elemento dell'attributo Nome è il primo attributo nell'oggetto di business Accessory.

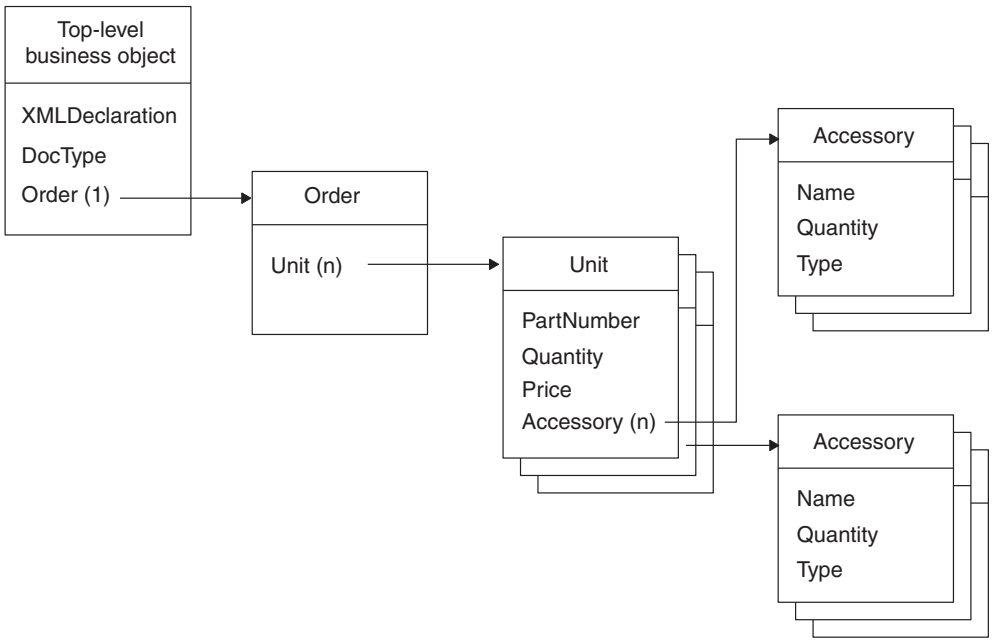

Figura 13. Esempio di oggetto di business per il documento XML utilizzando il DTD Order

Proprietà dell'attributo oggetto di business per DTD

Quando le definizioni oggetto di business per un documento XML sono basate sui DTD, le proprietà dell'attributo oggetto di business presentano le restrizioni discusse in "Proprietà dell'attributo oggetto di business" a pagina 34. Inoltre, la sintassi DTD può determinare la obbligatorietà di un attributo di un oggetto di business. L'obbligatorietà è una combinazione di fattori, inclusa la cardinalità e se l'attributo è una chiave, che determina se il gestore dati XML richiede l'attributo. se è richiesto un attributo, la relativa proprietà Required dell'attributo deve essere impostata su true.

L'impostazione della proprietà dell'attributo Required dipende dall'elemento XML e dalle specifiche dell'attributo, oltre alle impostazioni delle proprietà degli attributi Cardinalità, Key e Foreign Key, come di seguito spiegato:

- La cardinalità di un attributo dell'oggetto di business è determinata dal frammento `ELEMENT` nel DTD. Tale cardinalità riguarda se l'attributo è richiesto. Tabella 13 descrive la cardinalità e l'obbligatorietà per combinazioni possibili di dichiarazioni di un elemento in un DTD.

Tabella 13. Cardinalità e "Obbligatorietà" per un DTD

frammento DTD ELEMENT	Cardinalità	Required
Nessuno specificato	1	Sì
?	1	No
+	N	Sì
*	N	No

- Se un attributo dell'oggetto di business è una principale o la chiave esterna è determinata dal frammento `ATTLIST` nel DTD. La presenza di una chiave riguarda se l'attributo è richiesto. Tabella 14 descrive come la sintassi nel frammento `ATTLIST` interessa l'obbligatorietà dell'attributo dell'oggetto di business per combinazioni possibili delle dichiarazioni dell'attributo in un DTD.

Tabella 14. Chiavi e Obbligatorietà per un DTD

frammento DTD ATTLIST	Key	Required	Commento
#IMPLIED	No	No	
#REQUIRED	No	Sì	
ID {#IMPLIED | #REQUIRED}	Sì	No	#IMPLIED, #REQUIRED ignorato
IDREF {#IMPLIED, #REQUIRED}	Foreign key	Dipende se viene specificato #IMPLIED o #REQUIRED	
NMTOKEN {#IMPLIED | #REQUIRED}	Sì	No	#IMPLIED, #REQUIRED ignorato

Informazioni specifiche sull'applicazione per componenti XML nei DTD

Questa sezione fornisce le seguenti informazioni sul formato delle informazioni specifiche sull'applicazione per le definizioni dell'oggetto di business in base ai DTD:

- "Informazioni specifiche sull'applicazione del livello dell'oggetto di business"
- "Informazioni specifiche sull'applicazione dell'attributo Array" a pagina 48
- "Informazioni specifiche sull'applicazione dell'attributo" a pagina 49

Informazioni specifiche sull'applicazione del livello dell'oggetto di business: Il gestore dati XML utilizza i seguenti tipi di oggetti di business per rappresentare differenti generi di elementi XML generati da un DTD:

- "Definizioni di oggetto di business regolare in base ai DTD"
- "Definizioni di oggetti di business misti in base ai DTD"
- "Definizioni dell'oggetto di business wrapper in base ai DTD" a pagina 46

Questi tipi di oggetti di business sono distinti dalle informazioni specifiche sull'applicazione al livello di oggetto di business.

Definizioni di oggetto di business regolare in base ai DTD: Un oggetto di business *regolare* rappresenta un elemento XML. In questo tipo di oggetto di business, le informazioni specifiche sull'applicazione al livello di oggetto di business identificano il nome dell'elemento XML che l'oggetto di business rappresenta. Ad esempio, supporre che l'elemento XML sia definito come:

```
<!ELEMENT Unit(...)>
```

Le informazioni specifiche sull'applicazione a livello dell'oggetto di business per la definizione dell'oggetto di business associato è:

```
[BusinessObjectDefinition]
Name = MyApp_Unit
AppSpecificInfo = elem_name=Unit
[Attribute]
...
```

Definizioni di oggetti di business misti in base ai DTD: Un oggetto di business *misto* rappresenta un elemento XML misto, contenente un contenuto misto di dati carattere (#PCDATA) e altri sottoelementi. La rappresentazione DTD di un elemento XML di tipo misto è la seguente:

```
<!ELEMENT (#PCDATA | CONTAINED_ELEMENT1 | CONTAINED_ELEMENTN)*>
```

Ad esempio, supporre che l'elemento XML Cust sia definito nel DTD nella modo seguente:

```
<!ELEMENT Cust(#PCDATA | Address | Phone)*>
```

Per rappresentare un elemento XML di tipo misto, utilizzare una definizione oggetto di business di tipo misto. Per una definizione di oggetto di business misto, le relative informazioni specifiche sull'applicazione del livello dell'oggetto di business sono costituite dai seguenti componenti:

- Il nome dell'elemento XML misto
- La tag type=MIXED

Per la definizione di oggetto di business, MyApp_Cust, che rappresenta l'elemento Cust, le informazioni specifiche sull'applicazione al livello di oggetto di business sono rappresentate come segue:

```
[BusinessObjectDefinition]
Name = MyApp_Cust
AppSpecificInfo = Cust;type=MIXED;
```

Definizioni dell'oggetto di business wrapper in base ai DTD: Un oggetto di business wrapper rappresenta un elenco di scelte ripetute. Questo tipo di definizione di oggetto di business è richiesto quando un elemento XML ha elementi secondari che possono apparire in qualsiasi ordine e può essere di qualsiasi cardinalità. Un oggetto di business wrapper conserva l'ordine e la cardinalità degli elementi secondari nell'elemento XML.

Per un elemento XML elenco di scelta, la definizione DTD sarà:

```
(CONTAINEDELEMENT1 | ... | CONTAINEDELEMENTN)*
```

Ad esempio, la definizione dell'elemento XML elenco di scelta in un DTD potrebbe essere:

```
<!ELEMENT CUST( U | I | B )* )>
```

Questo elemento contiene tre sottoelementi che sono opzionali e che possono apparire in qualsiasi ordine. Ogni sottoelemento è un elemento semplice. Figura 14 mostra un documento XML di questo tipo.

```
<CUST>
   <U>.....
   </U>
   <B>.....
   </B>
   <I>.....
   </I>
   <B>.....
   </B>
   <U>.....
   </U>
...
```

Figura 14. Contenuto del documento XML per un elenco di scelte ripetute

Per rappresentare un elemento XML elenco di scelta definito in un DTD, la definizione dell'oggetto di business è gerarchica. Include le seguenti definizioni di oggetto di business:

- La definizione di oggetto di business principale

  Questo oggetto di business principale contiene un singolo attributo che rappresenta un array dell'oggetto di business a cardinalità multipla. Questo

attributo ha come il relativo tipo la definizione di oggetto di business per l'oggetto di business wrapper associato. La definizione di oggetto di business principale contiene le seguenti informazioni specifiche sull'applicazione:

- Al livello di oggetto di business, la definizione di oggetto di business principale contiene il nome dell'elemento choice-list nelle relative informazioni specifiche sull'applicazione
- Al livello di attributo, l'attributo a cardinalità multipla (nella definizione di oggetto di business principale) specifica gli elementi di scelta facoltativi nel seguente formato:

  (*choiceElement1*|...|*choiceElementN*)

  dove *choiceElement1*...*choiceElementN* corrisponde ad ognuno degli elementi scelta definiti. E' necessario che il carattere pipe (|) separi ciascuno degli elementi di scelta e l'intera tag si racchiusa da parentesi.

- Definizione di oggetto di business wrapper

  La definizione di oggetto di business wrapper contiene un attributo per ciascuno degli elementi di scelta definiti nell'elemento elenco di scelta. L'elemento *non* richiede informazioni specifiche sull'applicazione al livello di oggetto di business.

Figura 15 mostra un'immagine della gerarchia delle definizioni dell'oggetto di business per un elemento XML elenco di scelta. Durante l'esecuzione, ogni oggetto di business secondario è un'istanza di un oggetto di business wrapper e ha solo un attributo compilato con i dati. Ad esempio, un oggetto di business per il contenuto XML in Figura 14 a pagina 46 dovrebbe avere cinque elementi secondari, ciascuno con l'attributo appropriato compilato.

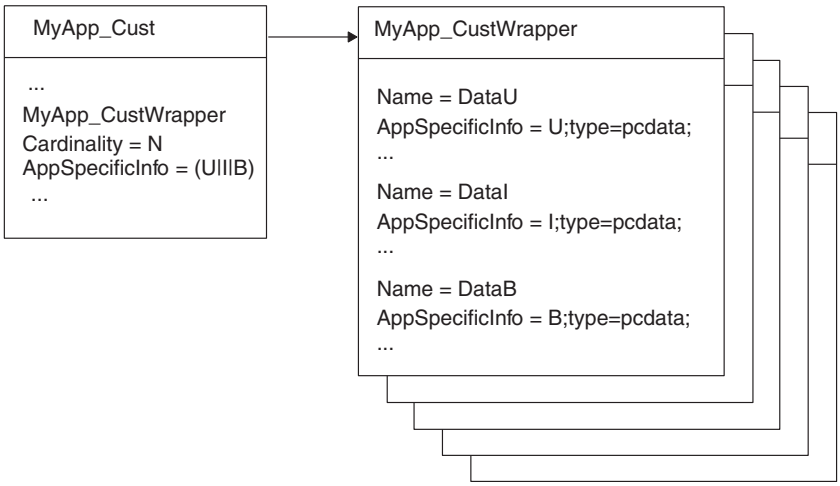

Figura 15. Definizione di oggetto di business gerarchico per un elemento XML elenco di scelta

Figura 16 a pagina 48 mostra la definizione dell'oggetto di business principale, `MyApp_Cust`, con le relative informazioni specifiche sull'applicazione.

```
[BusinessObjectDefinition]
Name = MyApp_Cust
AppSpecificInfo =

   [Attribute]
   Name = CustWrapper
   Type = MyApp_CustWrapper
   Cardinality = N
   AppSpecificInfo = attr_name=CustWrapper;(U|I|B)
[End]
```

Figura 16. Definizione di oggetto di business principale per un elemento elenco di scelta

La definizione di oggetto di business wrapper, `MyApp_CustWrapper`, ha tre attributi, uno per ogni elemento scelta. Poiché ogni elemento di scelta contiene dati di carattere, le informazioni specifiche sull'applicazione per ciascun attributo specifica:

- Il nome dell'elemento
- La tag `type=pcdata`

Nota: Per ulteriori informazioni sugli attributi per i dati di carattere, fare riferimento a "Per un elemento XML con solo PCDATA" a pagina 50.

Figura 17 mostra la definizione oggetto di business wrapper per questo documento XML.

```
[BusinessObjectDefinition]
Name = MyApp_CustWrapper
AppSpecificInfo =

   [Attribute]
   Name = DataU
   Type = String
   AppSpecificInfo = attr_name=U;type=pcdata;
   [End]

   [Attribute]
   Name = DataI
   Type = String
   AppSpecificInfo = attr_name=I;type=pcdata;
   [End]

   [Attribute]
   Name = DataB
   Type = String
   AppSpecificInfo = attr_name=B;type=pcdata;
   [End]
```

Figura 17. definizione oggetto di business wrapper per un elemento elenco di scelta

Informazioni specifiche sull'applicazione dell'attributo Array: Se un attributo di un oggetto di business rappresenta un elemento XML contenente altri elementi, le informazioni specifiche sull'applicazione devono contenere il nome dell'elemento. Ad esempio, se un attributo chiamato `DeliveryDate` ha un tipo di oggetto di business e rappresenta un elemento chiamato DATETIME, le informazioni specifiche sull'applicazione contengono il nome dell'elemento:

```
Name = DeliveryDate
Relationship = Containment
Cardinality = n
AppSpecificInfo = DATETIME
```

Informazioni specifiche sull'applicazione dell'attributo: L'attributo della definizione di un oggetto di business può rappresentare i seguenti componenti XML:

- "Per elementi XML"
- "Per un elemento XML con solo PCDATA" a pagina 50
- "Per un attributo XML" a pagina 50
- "Per un elemento XML con dati di carattere e attributi" a pagina 51
- "Per un elemento o attributo XML che contenga caratteri speciali" a pagina 52
- "Per una dichiarazione XML DOCTYPE" a pagina 52
- "Per una sezione CDATA" a pagina 53
- "Per un commento XML" a pagina 53
- "Per istruzioni di elaborazione XML" a pagina 54

Tabella 25 mostra le tag per le informazioni specifiche sull'applicazione al livello di attributo per questi differenti componenti XML insieme alle sezioni di questo manuale che descrivono queste tag con maggiori dettagli.

Tabella 15. Tag per le informazioni specifiche sull'applicazione dell'attributo

Rappresentazione dell'attributo dell'oggetto di business	Informazioni specifiche sull'applicazione	Per ulteriori informazioni
Un elemento XML	`elem_name=`*`nome dell'elemento XML`*	"Per elementi XML"
Un elemento XML con solo PCDATA	`elem_name=`*`nome dell'elemento XML`*`;type=pcdata`	"Per un elemento XML con solo PCDATA" a pagina 50
Un attributo per un elemento XML	`attr_name=`*`nome dell'attributo XML`* `type=attribute`	"Per un attributo XML" a pagina 50
Un elemento XML contenente dati di carattere e attributi	`type=pcdata;notag`	"Per un elemento XML con dati di carattere e attributi" a pagina 51
Un elemento XML o un attributo il cui contenuto include caratteri speciali	`escape=true`	"Per un elemento o attributo XML che contenga caratteri speciali" a pagina 78
Per una dichiarazione DOCTYPE	`type=doctype`	"Per una dichiarazione XML DOCTYPE" a pagina 52
Per una sezione CDATA	`type=cdata`	"Per una sezione CDATA" a pagina 53
Un commento da aggiungere al documento XML	`type=comment`	"Per un commento XML" a pagina 78
Un'istruzione di elaborazione	`type=pi`	"Per le istruzioni di elaborazione XML" a pagina 78

Nota: Le informazioni specifiche sull'applicazione dell'attributo possono anche includere una tag con il formato `( a | b | c )` per specificare un attributo a cardinalità multipla che rappresenta una scelta ripetuta. Per ulteriori informazioni, consultare "Definizioni dell'oggetto di business wrapper in base ai DTD" a pagina 46.

Per elementi XML: Ogni semplice (`String`) attributo dell'oggetto di business che rappresenta un elemento XML deve includere la tag `elem_name` nelle relative informazioni specifiche sull'applicazione per identificare l'elemento associato:

`elem_name=`*`nome dell'elemento XML`*

Ad esempio, se un attributo di un oggetto di business CustLName rappresenta un attributo semplice XML, le relative informazioni specifiche sull'applicazione sono:

```
Name = CustLName
AppSpecificInfo = elem_name=CustLName;
```

I nomi elemento XML può contenere caratteri speciali (ad esempio punti e trattini). Tuttavia, i nomi degli attributi dell'oggetto di business non possono contenere questi caratteri speciali. Perciò, il nome dell'elemento XML deve essere specificato nella tag elem_name. Per denominare l'attributo dell'oggetto di business, l'ODA XML rimuove qualsiasi carattere speciale nel nome dell'elemento XML, sostituendoli con il carattere sottolineatura(_).

Nell'esempio seguente, le informazioni specifiche sull'applicazione per l'elemento XML specificano un nome differente dall'attuale nome dell'elemento XML poiché l'attributo contiene un carattere speciale:

```
Name = Phone_Tag
AppSpecificInfo = elem_name=Phone#Tag;
```

Il nome attuale dell'elemento XML contiene il carattere cancelletto (#), che non è valido nei nomi degli attributi dell'oggetto di business. Perciò, la tag elem_name nelle informazioni specifiche sull'applicazione specificano il nome attuale dell'elemento XML. Nel nome dell'attributo associato dell'oggetto di business, il carattere cancelletto è sostituito con il carattere sottolineatura.

Per un elemento XML con solo PCDATA: Gli elementi XML che contengono solo dati carattere sono elementi misti, che contengono solo il programma di specifica del contenuto dell'elemento PCDATA. Un attributo di oggetto di business che rappresenta un elemento XML con solo PCDATA deve avere la seguente tag type nelle informazioni specifiche sull'applicazione:

```
type=pcdata
```

In questo caso, il nome dell'elemento è il primo campo nelle informazioni specifiche sull'applicazione e il parametro type è il secondo campo.

Ad esempio, un elemento chiamato PartNumber contenente solo PCDATA dovrebbe avere la seguente definizione nel DTD:

```
<!ELEMENT PartNumber (#PCDATA)>
```

Il corrispondente attributo nella definizione dell'oggetto di business dovrebbe avere le seguenti informazioni specifiche sull'applicazione:

```
Name = PartNumber
AppSpecificInfo = elem_name=PartNumber;type=pcdata;
```

Se le informazioni specifiche sull'applicazione contengono anche il testo notag, il gestore dati XML non genera markup XML. Aggiunge solo il valore dell'attributo stesso al documento XML. Per ulteriori informazioni, consultare "Per un elemento XML con dati di carattere e attributi" a pagina 51.

Per un attributo XML: Se un attributo di un oggetto di business rappresenta un attributo di un elemento XML, le relative informazioni specifiche sull'applicazione devono includere le seguenti tag:

- La tag attr_name:

  attr_name=*attrName*

  I nomi attributo XML possono contenere caratteri speciali (ad esempio punti e trattini). Tuttavia, i nomi degli attributi dell'oggetto di business non possono

contenere questi caratteri speciali. Perciò, il nome dell'attributo XML deve essere specificato nella tag attr_name. Per denominare l'attributo dell'oggetto di business, l'ODA XML rimuove qualsiasi carattere speciale nel nome dell'attributo XML.

- La tag type:

  type=attribute

  Questa tag type identifica lo scopo dell'attributo dell'oggetto di business associato come un attributo XML.

Nota: Come descritto in "Struttura oggetto di business" a pagina 33, tutti gli attributi dell'oggetto di business che rappresentano attributi XML devono trovarsi nelle definizioni dell'oggetto di business *prima* di ogni attributo dell'oggetto di business che rappresenti elementi XML.

Ad esempio, se un attributo di un oggetto di business chiamato ID rappresenta un attributo XML chiamato ID, le relative informazioni specifiche sull'applicazione sono:

```
Name = ID
AppSpecificInfo = attr_name=ID;type=attribute;
```

Per un altro esempio che utilizza la tag type=attribute, fare riferimento a "Per un elemento XML con dati di carattere e attributi".

Per un elemento XML con dati di carattere e attributi: Se un elemento XML contiene *solo* PCDATA o CDATA e ha uno o più attributi XML, la relativa definizione di oggetto di business deve includere i seguenti attributi:

- Un attributo dell'oggetto di business per ogni attributo XML.

  Il nome attributo deve corrispondere al nome dell'attributo XML. Le relative informazioni specifiche sull'applicazione del livello attributo devono includere le tag attr_name e type=attribute. Per ulteriori informazioni, consultare "Per un attributo XML" a pagina 50.
- Un attributo oggetto di business per i dati di carattere associati al programma di specifica del contenuto dell'elemento PCDATA o CDATA.

  Questo attributo contiene i dati che sono associati all'elemento principale XML. Le relative informazioni specifiche sull'applicazione devono contenere:
  - La tag appropriata type=*typename*, (dove *typename* è o pcdata o cdata) seguita dal punto e virgola (;)
  - Lo script parola chiave notag, che impedisce al gestore dati XML di generare tag di avvio duplicate (una per l'oggetto di business e una per l'attributo). Il gestore dati XML crea una tag XML di avvio e di fine per ogni attributo dell'oggetto di business *a meno che* notag non appaia nelle informazioni specifiche sull'applicazione per questo attributo.

Ad esempio, supporre di avere un elemento XML chiamato Price che abbia un attributo chiamato Currency e che richieda dati per Price:

```
<!ELEMENT Price (#PCDATA)>
<!ATTLIST Price Currency NMTOKEN #IMPLIED>
```

Poiché l'elemento Price ha un attributo XML, nella relativa definizione dell'oggetto di business deve essere creato un attributo dell'oggetto di business per Currency. Inoltre, deve esistere un altro attributo per contenere i dati Price. L'attributo per i dati Price deve specificare notag nelle relative informazioni specifiche sull'applicazione per impedire al gestore dati di creare una tag di avvio e di fine per questo attributo.

L'oggetto di business secondario Price dovrebbe essere simile a:

```
[BusinessObjectDefinition]
Name = Price
AppSpecificInfo = Price
   [Attribute]
   Name = Currency
   Type = String
   AppSpecificInfo = attr_name=Currency;type=attribute;
   ...
   [End]
   [Attribute]
   Name = Price
   Type = String
   AppSpecificInfo = Price;type=pcdata;notag
   ...
   [End]
```

In questo caso, il gestore dati *non* genera un nuovo elemento XML per i dati Price ma aggiunge semplicemente i dati all'elemento principale.

Per un elemento o attributo XML che contenga caratteri speciali: Attributi dell'oggetto di business che rappresentano XML elementi o XML attributi con un contenuto che richiede elaborazione escape deve includere la seguente tag nelle relative informazioni specifiche sull'applicazione:

```
escape=true
```

Un attributo richiede l'elaborazione escape se l'attributo rappresenta un elemento XML il cui valore contiene uno qualsiasi dei seguenti caratteri speciali:

- virgoletta singola (')
- virgolette doppie (")
- e commerciale (&)
- minore di (<)
- maggiore di (>)

Un attributo *non* non verrà elaborato via escape a meno che non contenga la tag escape=true nelle relative informazioni specifiche sull'applicazione. Questa tag deve essere posizionata alla fine di qualsiasi informazione specifica sull'applicazione esistente. Ad esempio:

```
[Attribute]
Name=Data
Type=String
AppSpecificInfo=Price;type=pcdata;escape=true
[End]
```

se le applicazioni specifiche sull'applicazione dell'attributo *non* includono la tag escape, il gestore dati XML controlla il valore proprietàDefaultEscapeBehavior per determinare se effettuare l'elaborazione escape:

- Se DefaultEscapeBehavior è impostato su true, il gestore dati XML esegue l'elaborazione escape su *tutti* i valori dell'attributo.
- Se DefaultEscapeBehavior è impostato su false, il gestore dati XML eseguirà *solo* l'elaborazione escape sugli attributi le cui informazioni specifiche sull'applicazione contengono la tag escape.

Per una dichiarazione XML DOCTYPE: Se un attributo di un oggetto di business rappresenta un tipo di documento dichiarazione nel prologo, le informazioni specifiche sull'applicazione devono includere la seguente tag type:

```
type=doctype
```

Ad esempio, se un attributo di un oggetto di business chiamato `DocType` rappresenta un elemento DOCTYPE, le relative informazioni specifiche sull'applicazione saranno:

```
Name = DocType
AppSpecificInfo = type=doctype;
```

Se l'attributo `DocType` ha il valore:

```
DOCTYPE CUSTOMER "customer.dtd"
```

allora il gestore dati genera il seguente elemento XML:

```
<!DOCTYPE CUSTOMER "customer.dtd">
```

Inoltre, queste informazioni specifiche sull'applicazione possono essere utilizzate per includere dichiarazioni di entità generali nel documento XML. Tuttavia, non è presente un supporto specifico per l'inclusione di un DTD interno o di un parametro dichiarazioni entità. Questi possono essere inclusi nel documento inserendo l'intero testo nel valore di un attributo che presenti `type=doctype` nelle informazioni specifiche sull'applicazione.

Per una sezione CDATA: Durante la conversione di un documento XML in un oggetto di business, il gestore dati XML analizza il contenuto nella sezione `CDATA`. Il gestore dati ritiene che il contenuto deve essere un'istanza XML.

Se un attributo di un oggetto di business rappresenta una sezione `CDATA` con un documento XML, le informazioni specifiche sull'applicazione devono includere la seguente tag `type`:

```
type=cdata
```

Ad esempio, se un attributo dell'oggetto di business `UserArea` rappresenta una sezione `CDATA`, le relative informazioni specifiche sull'applicazione saranno:

```
Name = UserArea
AppSpecificInfo = type=cdata;
```

Per un commento XML: Quando il gestore dati XML converte un oggetto di business in un documento XML, è possibile specificare che il gestore dati aggiunge commenti al documento XML. Per abilitare il gestore dati ad aggiungere commenti, procedere nel modo seguente:

- Creare nella definizione di oggetto di business l'attributo (o gli attributi) dell'oggetto di business che rappresentano i commenti.

  Nota: L'ODA XML *non* genera automaticamente attributi dell'oggetto di business per i commenti XML. E' necessario aggiungere manualmente tali attributi come descritto in "Creazione manuale delle definizioni dell'oggetto di business" a pagina 83.
- Includere la seguente tag `type` al livello attributo nelle informazioni specifiche sull'applicazione:

  ```
  type=comment
  ```
- Nell'oggetto di business attuale, specificare testo di commento come valore per tale attributo.

Ad esempio, se un attributo di un oggetto di business chiamato `Comment` rappresenta un documento XML che il gestore dati dovrebbe aggiungere ad un documento XML, l'attributo `Comment` dovrebbe essere il seguente:

```
Name = Comment
AppSpecificInfo = type=comment;
```

Se tale attributo ha il valore "Informazioni cliente aggiornate da applicazione A", viene generato il seguente XML:

```
<!--Informazioni cliente aggiornate dall'applicazione A-->
```

Per istruzioni di elaborazione XML: Se un attributo di un oggetto di business rappresenta una istruzione di elaborazione, le informazioni specifiche sull'applicazione devono includere la seguente tag type:

```
type=pi
```

Ad esempio, se un attributo di un oggetto di business chiamato XMLDeclaration rappresenta la dichiarazione XML nel prologo, le informazioni specifiche sull'applicazione saranno:

```
Name = XMLDeclaration
AppSpecificInfo = type=pi;
```

Se l'attributo ha il valore:

```
xml version = "1.0"
```

allora il gestore dati XML genera il seguente XML:

```
<?xml version="1.0"?>
```

Creazione di definizioni di un oggetto di business dai DTD

Un DTD descrive il formato di un documento XML. Perciò, il DTD è molto utile per ottenere le informazioni richieste per la definizione di un oggetto di business. Per tradurre le informazioni della struttura nel DTD ad una definizione di un oggetto di business, è possibile utilizzare XML Object Discovery Agent (ODA). Per ulteriori informazioni su ODA XML, fare riferimento a "Utilizzo di ODA XML per creare definizioni degli oggetti di business" a pagina 83.

Nota: Le precedenti versioni del gestore dati XML includevano due strumenti per creare definizioni dell'oggetto di business dai DTD: Edifecs SpecBuilder e la deprecata utilità XMLBorgen. L'ODA XML sostituisce entrambi questi strumenti con una maggiore funzionalità.

Strutture DTD supportate

L'ODA XML supporta le strutture di un DTD incluso:

- Entità — entità dei seguenti formati definiti dall'utente sono riconosciuti e sostituiti dovunque sia riferito:

  ```
  <!ENTITY % name value>
  ```

  Nota: L'ODA XML rimuove i caratteri newline (\n), caratteri di ritorno a capo (\r) o i caratteri di tabulazione (\t) se i caratteri sono presenti tra la tag ENTITY e il relativo contenuto o tra un elemento o la tag attributo e il contenuto della tag.
- Esterni DTD — ODA XML supporta riferimenti a DTD esterni (dove un DTD si riferisce ad uno o più DTD). Possono risolvere DTD esterni *solo* se si trovano sul file system locale; non possono accedere ad una URL per cercarli. Entrambi questi strumenti cercano sempre di trovare riferimenti ai DTD esterni sul file system locale; *non* li ignorano.
- direttiva ANY — L'ODA XML crea un attributo String per un elemento con ANY nel suo contenuto. Ad esempio, se il DTD ha il seguente costrutto:

```
<!ELEMENT SCENE (ANY) >
```

La corrispondente definizione di oggetto di business sarà:

```
[Attribute]
Name = SCENE
Type = String
Cardinality = 1
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = true
AppSpecificInfo = SCENE;type=pcdata;
[End]
```

- Prologo — Il gestore dati XML compila le informazioni del prologo come `DOCTYPE` e dichiarazione XML, finché la definizione di oggetto di business contiene gli attributi che corrispondono a quegli elementi. Tuttavia, il gestore dati compila *solo* il nome di `DOCTYPE`; *non* compila altre meta informazioni possibili.

Strutture DTD non supportate

L'ODA XML è in grado di elaborare la maggior parte di DTD. Tuttavia, *non* supporta le seguenti strutture DTD:

- Sezioni condizionali — Tali sezioni presentano la struttura seguente:

  ```
  <![INCLUDE[ informazioni che devono essere incluse ]]>
  <![IGNORE[ informazioni che devono essere ignorate ]]>
  ```

- Namespaces — Tag del formato `xxx:yyy` è elaborata semplicemente come una tag `xxx:yyy` e non come una tag `yyy` che appartiene ad uno spazio dei nomi `xxx`.
- La parola chiave `EMPTY` — L'ODA XML genererà un oggetto di business senza attributi se incontra un elemento che utilizza questa parola chiave. Gli utenti possono risolvere questo aggiungendo un attributo fittizio alla definizione dell'oggetto di business controllando il relativo indicatore "Chiave". Per ulteriori informazioni riguardo agli attributi chiave, fare riferimento a "Proprietà degli attributi Chiave e Chiave esterna" a pagina 35.

Documenti XML che utilizzano documenti dello schema

Uno *Schema XML* è un modello di dati per documenti XML che utilizza un *documento dello schema* (estensione`.xsd`) per definire la maschera (schema) di un documento XML. Diversamente da un DTD, un documento dello schema utilizza la stessa sintassi di un documento XML per descrivere lo schema. Le definizioni dell'oggetto di business che rappresentano lo schema di un documento XML utilizzano le informazioni nel documento dello schema per conservare e registrare la struttura del documento. Questa sezione fornisce le seguenti informazioni riguardo le informazioni della struttura derivante per una definizione di un oggetto di business da un documento dello schema:

- "Requisiti per le definizioni di un oggetto di business in base ai documenti dello schema"
- "Creazione delle definizioni di business da documenti dello schema" a pagina 80

Requisiti per le definizioni di un oggetto di business in base ai documenti dello schema

Per assicurarsi che le definizioni di un oggetto di business che rappresenta documenti dello schema siano conformi ai requisiti del gestore dati XML, utilizzare le linee guida in questa sezione, che comprende:

- "Struttura di un oggetto di business per documenti dello schema" a pagina 56

- "Proprietà dell'attributo di un oggetto di business per documenti dello schema" a pagina 64
- "Informazioni specifiche sull'applicazione per componenti XML nei documenti dello schema" a pagina 65

Struttura di un oggetto di business per documenti dello schema

Un oggetto di business che viene elaborato dal gestore dati XML utilizzando le definizioni dell'oggetto di business in base ai documenti dello schema deve deve obbedire alle seguenti regole:

- Il documento dello schema deve avere le seguenti definizioni dell'oggetto di business richieste:
  - Una definizione dell'oggetto di business di livello superiore rappresenta l'elemento `schema`.
  - Una definizione dell'oggetto di business elemento root rappresenta l'elemento root del documento dello schema. Una definizione di un oggetto di business regolare, misto o wrapper rappresenta l'elemento root nel documento dello schema.

  Per ulteriori informazioni, consultare "Definizioni dell'oggetto di business richieste per i documenti dello schema" a pagina 57.
- Una definizione di un oggetto di business regolare, misto o wrapper può rappresentare anche componenti XML contenuti.

Un esempio di documento dello schema per un documento XML è mostrato in Figura 18. Il documento dello schema è denominato `Order` e contiene elementi che corrispondono ad una entità di applicazione `Order`. Tale esempio di documento dello schema descrive la stessa struttura dell'oggetto di business che descrive il documento di esempio DTD. Figura 13 a pagina 44 mostra la struttura di una definizione di un oggetto di business che potrebbe essere creata per corrispondere ad un documento XML associato al documento dello schema `Order`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">
        <xs:complexType name="AccessoryType">
                <xs:sequence>
                        <xs:element ref="Quantity"/>
                        <xs:element ref="Type"/>
                </xs:sequence>
                <xs:attribute name="Name" type="xs:string" use="required"/>
        </xs:complexType>
        <xs:element name="Order">
                <xs:complexType>
                        <xs:sequence>
                                <xs:element name="Unit" type="UnitType"
                    maxOccurs="unbounded"/>
                        </xs:sequence>
                </xs:complexType>
        </xs:element>
        <xs:element name="PartNumber" type="xs:string"/>
        <xs:element name="Price" type="xs:string"/>
        <xs:element name="Quantity" type="xs:string"/>
        <xs:element name="Type" type="xs:string"/>
        <xs:complexType name="UnitType">
                <xs:sequence>
                        <xs:element ref="PartNumber" minOccurs="0"/>
            <xs:element ref="Quantity"/>
            <xs:element ref="Price"/>
            <xs:element name="Accessory" type="AccessoryType" minOccurs="0"
                   maxOccurs="unbounded"/>
                </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Figura 18. Esempio di documento dello schema XML

Definizioni dell'oggetto di business richieste per i documenti dello schema: Per effettuare una rappresentazione un documento dello schema richiede almeno le due definizioni dell'oggetto di business descritte in "Struttura oggetto di business" a pagina 33. Per un documento dello schema, tali definizioni dell'oggetto di business hanno i seguenti requisiti aggiuntivi:

- Il un oggetto di business*di livello superiore* rappresenta l'elemento di `schema` e deve contenere i seguenti attributi:
  - Un attributo denominato `XMLDeclaration` per rappresentare la versione XML

    Questo attributo deve avere la tag `type=pi` nelle relative informazioni specifiche sull'applicazione. Per ulteriori informazioni, consultare "Per le istruzioni di elaborazione XML" a pagina 78.
  - Un attributo per rappresentare l'elemento root nel documento dello schema

    Come descritto in "Struttura oggetto di business" a pagina 33, questo attributo deve avere come il relativo tipo un oggetto di business a cardinalità singola, il cui tipo sia la definizione di oggetto di business per l'elemento root. E' necessario che le informazioni specifiche sull'applicazione elenchino il nome di tale elemento con la tag `elem_name`. Per ulteriori informazioni, consultare "Per elementi XML" a pagina 73.
- Una definizione dell'oggetto di business *elemento root* rappresenta l'elemento root del documento dello schema. Poiché un documento dello schema può avere diversi componenti XML definiti a livello globale, è possibile indicare a ODA XML quale elemento deve considerare l'elemento root attraverso la sua proprietà di configurazione `Root`. La definizione dell'oggetto di business elemento root può contenere i seguenti attributi:

– Un attributo (facoltativo) schemaLocation per rappresentare dove il risiede il documento (o i documenti) di schema

  Se l'ODA XML genera un attributo schemaLocation nella definizione dell'oggetto di business di livello superiore questo dipende dalle impostazioni della relativa proprietà DocTypeOrSchemaLocation. Per ulteriori informazioni, consultare "Per le posizioni dello schema XML" a pagina 79.
– Un attributo per ognuno dei componenti XML nell'elemento root

Nota: Per un elenco dei requisiti di un oggetto di business generale, fare riferimento a "Requisiti per le definizioni oggetto di business" a pagina 33.

Entrambe queste definizioni necessarie dell'oggetto di business richiedono informazioni specifiche sull'applicazione al livello di oggetto di business che definiscono lo spazio dei nomi di destinazione e qualsiasi requisito nome componente. Per ulteriori informazioni, consultare "Informazioni specifiche sull'applicazione all livello dell'oggetto di business" a pagina 65.

L'elemento schema XML in Figura 18 a pagina 57 è il seguente:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">
```

Figura 19 mostra la definizione di oggetto di business di livello superiore, TopLevel, che rappresenta questo elemento schema.

```
[BusinessObjectDefinition]
Name=TopLevel
AppSpecificInfo=elem_fd=qualified;attr_fd=unqualified
...
   [Attribute]
   Name=XMLDeclaration
   Type=String
   AppSpecificInfo=type=pi;
   ...
   [End]
   [Attribute]
   Name=Order
   Type=TopLevel_Order
   AppSpecificInfo=elem_name=Order;
   ...
   [End]
...
[End]
```

Figura 19. Esempio di definizione dell'oggetto di business di livello superiore

Se l'ODA XML ha generato una definizione dell'oggetto di business di livello superiore in Figura 19, le seguenti proprietà di configurazione dell'ODA dovrebbero essere così impostate:

Proprietà di configurazione dell'ODA	Valore della proprietà
BOPrefix	Non impostato
livello superiore	"Livello superiore"
Root	"Order"
DoctypeorSchemaLocation	true

Per un esempio di una definizione dell'oggetto di business di un elemento root, fare riferimento a Figura 26 a pagina 70.

Definizioni dell'oggetto di business regolare nei documenti dello schema: Una definizione di un oggetto di business *regolare* rappresenta uno qualsiasi dei seguenti costrutti XML:

- Un elemento XML contenente un tipo complesso (sia denominato che anonimo) contenente una `sequenza` di gruppi degli elementi secondari.

  Ogni elemento secondario nella `sequenza` di gruppi è rappresentata come un attributo nella definizione dell'oggetto di business. Per ulteriori informazioni, consultare "Per elementi XML all'interno di un tipo complesso" a pagina 74.

  Nota: Se un tipo complesso contiene un gruppo `scelta` o un gruppo `tutti`, deve essere rappresentato da una definizione di oggetto di business wrapper. Per ulteriori informazioni, consultare "Definizioni di un oggetto di business wrapper in base ai documenti dello schema" a pagina 61.
- Un elemento XML contenente attributi

In tali tipi di definizioni dell'oggetto di business, le informazioni specifiche sull'applicazione al livello di oggetto di business *non* richiedono alcun tipo di informazioni speciali. Gli attributi di una definizione di oggetto di business regolare rappresentano gli elementi definiti all'interno di un tipo complesso XML.

Nota: Per tipi complessi contenenti una `sequenza` di gruppi di elementi secondari, ogni elemento secondario nella sequenza di gruppi viene rappresentato come un attributo nella definizione di oggetto di business.

Ad esempio, supporre che un elemento XML denominato Unit venga definito come:

```
<xsd:element name="Unit">
    <xsd:complexType>
    ...
    </xsd:complexType>
</element>
```

La seguente definizione di oggetto di business rappresenta l'elemento Unit:

```
[BusinessObjectDefinition]
Name = MyApp_Unit
AppSpecificInfo =
   [Attribute]
   ...
```

Definizioni di oggetto di business misto in base ai documenti dello schema: Una definizione di oggetto di business *misto* rappresenta un elemento XML misto, che contiene un contenuto misto di dati carattere ed altri sottoelementi. Un documento dello schema descrive un elemento XML di tipo misto come un tipo complesso con l'attributo `mixed` impostato su `true`, come di seguito riportato:

```
<xsd:complexType mixed="true">
    <xsd:sequence>
        <xsd:element name="subElement1" type="subElementType"/>
        ...
    </xsd:sequence>
</xsd:complexType>
```

Nota: Gli oggetti di business misti, come descritto in questa sezione, sono utilizzati per un elenco di elementi ripetuti *e* dati carattere. Se il proprio elenco di scelta *non* contiene dati carattere, utilizzare un oggetto di business wrapper come descritto in "Nomi qualificati del componente" a pagina 70.

Poiché questo tipo complesso imposta l'attributo `mixed` su `true`, è possibile che contenga dati carattere in aggiunta a uno o più sottoelementi che definisce. Se l'attributo `mixed` è impostato su `false`, non sono consentiti dati carattere nel tipo complesso.

Ad esempio, Figura 20 mostra un elemento XML di tipo misto definito in un documento dello schema.

```
<xsd:complexType name="Cust" mixed="true">
    <xsd:sequence>
        <xsd:element name="Name"/>
        <xsd:element name="Address"/>
        <xsd:element name="Phone"/>
    </xsd:sequence>
</xsd:complexType>
```

Figura 20. Esempio di documento dello schema per un elemento XML di tipo misto

Per effettuare una rappresentazione un elemento XML ti tipo misto richiede le due seguenti definizioni di oggetto di business:

- La definizione di oggetto di business principale

  Questo oggetto di business principale contiene un singolo attributo che rappresenta un array dell'oggetto di business a cardinalità multipla. Questo attributo ha come il relativo tipo la definizione di oggetto di business per l'oggetto di business wrapper associato. La definizione dell'oggetto di business principale include le seguenti informazioni specifiche sull'applicazione:
  - Le informazioni specifiche sull'applicazione al livello di oggetto di business sono costituite dai seguenti componenti:
    - Il nome dell'elemento XML associato di tipo misto seguito da un punto e virgola (;)
    - La tag `type=MIXED`
  - L'attributo a cardinalità multipla presenta la seguente tag nelle sue informazioni specifiche sull'applicazione:

    (*mixedTypeElement*|*subElement1*|...|*subElementN*)

    dove:
    - *mixedTypeElement* è il nome dell'elemento XML associato di tipo misto
    - *subelement1*...*subElementN* corrispondono ad ognuno dei sottoelementi definiti nel tipo complesso

    Il carattere pipe (|) deve separare ognuno dei sottoelementi e l'intera tag deve essere racchiusa da parentesi.
- Definizione di un oggetto di business wrapper

  La definizione dell'oggetto di business wrapper contiene gli attributi per i dati misti:
  - Un attributo per i dati carattere, che richiede la tag `type=pcdata` (per indicare caratteri dati) così come la tag `notag` (per indicare che i dati sono associati con l'elemento corrente, non con un elemento separato).
  - Un attributo per ognuno dei sottoelementi definiti nell'elemento di tipo misto. Ogni attributo richiede la tag `type=pcdata` (per indicare che rappresenta un tipo semplice).

  Per ulteriori informazioni sulla tag `type=pcdata`, fare riferimento a "Per elementi XML all'interno di un tipo complesso" a pagina 74.

Nota: Tale definizione di oggetto di business wrapper *non* richiede informazioni specifiche sull'applicazione al livello di oggetto di business.

Se la definizione di oggetto di business, `MyApp_Cust`, rappresenta l'elemento di tipo misto `Cust` in Figura 20 a pagina 60, le relative informazioni specifiche sull'applicazione sono rappresentate come segue:

```
[BusinessObjectDefinition]
Name = MyApp_Cust
AppSpecificInfo = type=MIXED;

   [Attribute]
   Name=Cust_wrapper1
   Type=MyApp_CustWrapper1
   Cardinality=n
   AppSpecificInfo=(Cust|Address|Phone)
   ...
   [End]
```

Definizioni di un oggetto di business wrapper in base ai documenti dello schema: Una definizione di oggetto di business *wrapper* rappresenta un elenco di scelta ripetute. Questo tipo di definizione di oggetto di business è richiesto quando un elemento XML ha elementi secondari che possono apparire in qualsiasi ordine e può essere di qualsiasi cardinalità. Un oggetto di business wrapper conserva l'ordine e la cardinalità degli elementi secondari in un particolare documento XML.

Nota: Gli oggetti di business wrapper, come descritto in questa sezione, sono utilizzati per un elenco di scelte ripetute che contiene elementi, *non* per i caratteri dati. Se il proprio elenco di scelte contiene dati caratteri, utilizzare un oggetto di business misto. Per ulteriori informazioni, consultare "Definizioni di oggetto di business misto in base ai documenti dello schema" a pagina 59.

Un documento dello schema può descrivere un elemento XML elenco di scelta come un tipo complesso che contiene uno dei seguenti gruppi di modelli:

- Un gruppo di `scelta`, che definisce un elenco non ordinato di elementi in cui deve apparire solo un elemento:

  ```
  <xsd:complexType>
      <xsd:choice>
      ...
      </choice>
  </complexType>
  ```

  Per rappresentare un elemento XML con un tipo complesso che contenga un gruppo di `scelta`, il gestore dati XML prevede la stessa definizione di oggetto di business gerarchica che crea per un elemento XML elenco di scelta definito in un DTD:

  – Definizione di un oggetto di business principale che contenga un attributo singolo a *cardinalità multipla* il cui tipo sia l'oggetto di business wrapper
  – Una definizione di oggetto di business wrapper che contenga un attributo per ogni sottoelemento nel gruppo di `scelta`.

  Tali definizioni di oggetto di business contengono informazioni specifiche sull'applicazione al livello di oggetto di business nello stesso formato dell'elemento XML elenco di scelta per un DTD. Per ulteriori informazioni, consultare "Definizioni dell'oggetto di business wrapper in base ai DTD" a pagina 46.
- Un gruppo `tutto`, che definisce un elenco non ordinato di elementi in cui gli elementi possono apparire non più di una volta ciascuno:

```
<xsd:complexType>
    <xsd:all>
    ...
    </xsd:all>
</xsd:complexType>
```

Per rappresentare un elemento XML con un tipo complesso che contenga un gruppo `tutto`, Il gestore dati XML prevede una definizione di oggetto di business gerarchica con le seguenti definizioni dell'oggetto di business:

- Una definizione dell'oggetto di business principale che contenga un singolo attributo a *cardinalità multipla* il cui tipo sia l'oggetto di business wrapper
- Una definizione dell'oggetto di business wrapper che contenga un attributo per ogni sottoelemento nel gruppo `tutto`

  Tali definizioni dell'oggetto di business contengono informazioni specifiche sull'applicazione al livello di oggetto di business nello stesso formato dell'elemento XML elenco di scelta per un DTD. Per ulteriori informazioni, consultare "Definizioni dell'oggetto di business wrapper in base ai DTD" a pagina 46.

Per indicare la cardinalità, tali gruppi di modelli supportano limitazioni delle ricorrenze con gli attributi `minOccurs` e `maxOccurs`. Per ulteriori informazioni, consultare Tabella 18 a pagina 64.

Ad esempio, la definizione dell'elemento XML in un documento dello schema potrebbe essere:

```
<xsd:element name="CUST">
    <xsd:complexType>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
            <xsd:element ref="U"/>
            <xsd:element ref="I"/>
            <xsd:element ref="B"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
```

Questo elemento contiene tre sottoelementi che sono facoltativi (sebbene un sottoelemento fuori dall'elenco *debba* ricorrere) e che possono apparire in qualsiasi ordine. Figura 14 a pagina 46 mostra un documento XML di questo tipo. Figura 16 a pagina 48 mostra la definizione dell'oggetto di business principale per questo documento XML. Figura 17 a pagina 48 mostra la definizione dell'oggetto di business wrapper.

Sostituzione del tipo nelle definizioni dell'oggetto di business in base ai documenti dello schema: *Sostituzione del tipo* abilita i tipi derivati ad apparire al posto dei loro tipi base nelle istanze individuali del documento XML. Quando si verifica la sostituzione del tipo, un elemento corrispondente ad una dichiarazione con un tipo di dati può avere qualsiasi tipo di dati che lo estende o lo limita. Nella seguente definizione dello schema, ShirtType e HatType sono tipi derivati del tipo base ProductType:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="items" type="ItemsType"/>
<xsd:complexType name="ItemsType">
  <xsd:sequence>
            <xsd:element ref="product" maxOccurs="unbounded"/>
          </xsd:sequence>
</xsd:complexType>
<xsd:element name="product" type="ProductType"/>
<xsd:complexType name="ProductType">
    <xsd:sequence>
          <xsd:element name="number" type="xsd:string"/>
            <xsd:element name="name" type="xsd:string"/>
      </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ShirtType">
    <xsd:complexContent>
    <xsd:extension base="ProductType">
          <xsd:sequence>
                <xsd:element name="size" type="xsd:string"/>
                <xsd:element name="color" type="xsd:string"/>
          </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="HatType">
    <xsd:complexContent>
    <xsd:extension base="ProductType">
      <xsd:sequence>
                <xsd:element name="size" type="xsd:string"/>
              </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
</xsd:schema>
```

Figura 21. Schema esempio con la sostituzione del tipo

In base allo schema precedente, il seguente documento XML è valido:

```
<items xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<product>
      <number>999</number>
      <name>Special Seasonal</name>
</product>
<product xsi:type="ShirtType">
      <number>557</number>
      <name>Short-Sleeved Linen Blouse</name>
      <size>M</size>
      <color>blue</color>
</product>
<product xsi:type="HatType">
      <number>443</number>
      <name>Four-Gallon Hat</name>
      <size>L</size>
</product>
</items>
```

Figura 22. Tipi derivati in un documento XML

Dovunque ProductType si verifica, i suoi tipi derivati, ShirtType e HatType, indicati dall'attributo xsi:type, possono invece apparire. Per rappresentare documenti XML in cui si verifica la sostituzione del tipo, il gestore dati XML crea un oggetto di business wrapper come un attributo secondario del documento XML. Questo oggetto di business wrapper ha attributi secondari che corrispondono al tipo complesso (ProductType in Figura 21) e i relativi tipi derivati (ShirtType e

HatType). Tabella 16 e Tabella 17 mostra le definizioni dell'oggetto di business che dovrebbero essere generati dallo schema XML schema precedente.

Tabella 16. Definizioni di oggetto di business per ItemsType

Nome attributo	Tipo	Cardinalità	ASI
Product	ProductTypeWrapper	N	(product);typeSub=true

Tabella 17. Definizioni dell'oggetto di business per ProductTypeWrapper

Attributo	Tipo	Cardinalità	ASI
ProductType	ProductType	1	Elem_name=product; xsiType=ProductType
ShirtType	ShirtType	1	Elem_name=product; xsiType=ShirtType;
HatType	HatType	1	Elem_name=product; xsiType=HatType;

Proprietà dell'attributo di un oggetto di business per documenti dello schema

Quando le definizioni degli oggetti di business per un documento XML sono basate sui documenti dello schema, le proprietà dell'attributo dell'oggetto di business presentano le restrizioni discusse in "Proprietà dell'attributo oggetto di business" a pagina 34. Inoltre, la sintassi del documento dello schema può determinare l'obbligatorietà di un attributo di un oggetto di business. L'obbligatorietà è una combinazione di fattori, inclusa la cardinalità e se l'attributo è una chiave, che determina se il gestore dati XML richiede l'attributo. se è richiesto un attributo, la relativa proprietà Required dell'attributo deve essere impostata su vero.

L'impostazione della proprietà dell'attributo Required dipende dall'elemento XML e dalle specifiche dell'attributo, oltre alle impostazioni delle proprietà degli attributi Cardinality, Key e Foreign Key, come di seguito spiegato:

- La cardinalità di un attributo dell'oggetto di business è determinata dall'indicatore Ricorrenza nel documento dello schema. Tale cardinalità riguarda se l'attributo è richiesto. Tabella 18 descrive la cardinalità e l'obbligatorietà per combinazioni possibili di dichiarazioni di un elemento in un documento dello schema.

Tabella 18. Cardinalità e "Obbligatorietà" per un documento dello schema

Indicatore di ricorrenza del frammento dello schema	Cardinality	Required
None specified	1	Sì
maxOccurs > 1	N	Sì
maxOccurs = ″unbounded″	N	Sì
minOccurs=0	No effect	No
minOccurs>1	N	Sì

- Se un attributo di un oggetto di business è richiesto viene anche determinato dall'attributo `use` nel documento dello schema. Tabella 19 descrive l'obbligatorietà per possibili valori dell'attributo `use`.

Tabella 19. "Obbligatorietà" per un documento dello schema

Attributo di ricorrenza del frammento dello schema: use	Cardinality	Required
None specified	No effect	No
`use=required`	No effect	Sì

- Se un attributo dell'oggetto di business è una principale o la chiave esterna è determinata dall'attributo `id` nel documento dello schema. La presenza di una chiave riguarda se l'attributo è richiesto. Tabella 20 descrive come il valore dell'attributo `id` riguarda l'obbligatorietà dell'attributo dell'oggetto di business.

Tabella 20. Chiavi e obbligatorietà per un documento dello schema

Attributo del frammento dello schema: id	Key	Required	Comment
`id=ID`	Sì	No	

Informazioni specifiche sull'applicazione per componenti XML nei documenti dello schema

Questa sezione fornisce le seguenti informazioni sul formato delle informazioni specifiche sull'applicazione per le definizioni dell'oggetto di business in base ai documenti dello schema:

- "Informazioni specifiche sull'applicazione all livello dell'oggetto di business"
- "Informazioni specifiche sull'applicazione dell'attributo" a pagina 72

Informazioni specifiche sull'applicazione all livello dell'oggetto di business: Il gestore dati XML utilizza i seguenti tipi di definizioni dell'oggetto di business per rappresentare i differenti generi di elementi XML root definiti in un documento dello schema. Questi tipi di definizione dell'oggetto di business sono distinte dalle informazioni specifiche sull'applicazione al livello dell'oggetto di business.

Tabella 21. Tag per le informazioni specifiche sull'applicazione al livello dell'oggetto di business

Tag nelle informazioni specifiche sull'applicazione	Descrizione	Per ulteriori informazioni
`target_ns`	Specifica lo spazio dei nomi di destinazione del documento dello schema	"Spazio dei nomi dello schema"
`attr_fd`	Specifica se i nomi attributo sono qualificati o non qualificati.	"Nomi qualificati del componente" a pagina 70
`elem_fd`	Specifica se i nomi degli elementi dichiarati localmente sono qualificati o non qualificati.	"Nomi qualificati del componente" a pagina 70

Nota: Le informazioni specifiche sull'applicazione al livello di business possono anche includere la tag `type=MIXED`. Per ulteriori informazioni, consultare "Definizioni di oggetto di business misto in base ai documenti dello schema" a pagina 59.

Spazio dei nomi dello schema: Diversamente dai DTD, i documenti dello schema richiedono una definizione di almeno uno spazio dei nomi. Uno *spazio dei nomi*

fornisce un contesto per i nomi degli elementi, i tipi dell'elemento e gli attributi all'interno di un documento XML. Uno spazio dei nomi è un URI (Uniform Resource Identifier), che include HTTP, FTP, così come altri generi di percorsi. Tabella 22 mostra gli spazi dei nomi che un documento dello schema può dichiarare di essere in grado di risolvere i riferimenti tra i componenti dello schema.

Tabella 22. Spazio dei nomi XML

Spazio dei nomi	Descrizione	Nome	Prefisso comune
Spazio dei nomi dello schema XML	Definisce ogni componente utilizzato in XSDL (XML Schema Definition Language), come ad esempio `element`, `schema` e `simpleType`.	`http://www.w3.org/2001/XMLSchema`	`xsd`, `xs`
Spazio dei nomi dell'istanza dello schema XML	Definisce quattro attributi associati con l'istanza dello schema: `type`, `nil`, `schemaLocation`, `noNamespaceSchemaLocation`	`http://www.w3.org/2001/XMLSchema-instance`	`xsi`
Spazio dei nomi definiti dall'utente	Definisce ogni componente dichiarato o definito da una dichiarazione globale (come ad esempio `element`, `attribute`, `type` o `group`). **Nota:** Gli elementi dichiarati localmente possono o non possono utilizzare lo spazio dei nomi di destinazione. Per ulteriori informazioni, consultare "Nomi qualificati del componente" a pagina 70.	User-defined	User-defined

Nota: L'ODA XML supporta documenti dello schema con spazi dei nomi di destinazione multipli.

Ogni documento dello schema può dichiarare uno *spazio dei nomi di destinazione*, che identifica lo spazio dei nomi a cui i componenti globali (elementi, attributi, tipi o gruppi) appartengono, con la tag`targetNamespace`. Se l'elemento dello schema include la tag `targetNamespace`, ogni definizione dell'oggetto di business generata per questo documento dello schema deve contenere la tag `target_ns` nelle relative informazioni specifiche sull'applicazione per specificare lo spazio dei nomi di destinazione dichiarati per il documento XML:

`target_ns=`*`indirizzo URI per lo spazio dei nomi di destinazione`*

La tag `target_ns` deve esistere nelle informazioni specifiche sull'applicazione al livello dell'oggetto di business per *tutte* le definizioni dell'oggetto di business, come mostrato di seguito:

- Per la definizione di oggetto di business di livello superiore, il valore della tag `target_ns` specifica il valore che l'attributo `targetNamespace` dell'elemento XML `schema` specifica.
- Ogni definizione dell'oggetto di business che rappresenta un elemento XML deve anche includere la tag `target_ns` nelle relative informazioni specifiche sull'applicazione. Poiché ogni componente globale (come ad esempio un elemento, un attributo o un tipo) appartiene al proprio spazio dei nomi di destinazione del documento dello schema, le definizioni dell'oggetto di business che rappresentano questi componenti specificano anche lo spazio dei nomi di destinazione del documento dello schema.

Nota: Le precedenti versioni del gestore dati XML prevedevano che la definizione dell'oggetto di business di livello superiore avesse attributi per i prefissi dello spazio dei nomi e usasse le appropriate `type=defaultNS` e `tagtype=xmlns` nelle informazioni specifiche sull'applicazione al livello dell'attributo. Questo meccanismo per definire i prefissi dello spazio dei nomi è stato sostituito, sebbene il gestore dati XML continui a supportarlo per la compatibilità con le definizioni dell'oggetto di business esistenti. Le nuove definizioni dell'oggetto di business dovrebbero utilizzare la tag `target_ns` come descritto in questa sezione. L'ODA XML è stato modificato per utilizzare la tag `target_ns`.

Ad esempio, il documento dello schema in Figura 23 definisce lo spazio dei nomi dello schema XML (con il prefisso `xsd`) e uno spazio dei nomi di destinazione (che è lo spazio dei nomi predefinito).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.ibm.com/ns1" xmlns="http://www.ibm.com/ns1">
   <xsd:complexType name="TaxInfoType">
      <xsd:sequence>
         <xsd:element name="SSN" type="string">
         </xsd:element>
         <xsd:element name="State" type="string">
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="Customer">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="TaxInfo" type="TaxInfoType">
            </xsd:element>
            <xsd:element name="BillTo" type="xsd:string">
            </xsd:element>
         </xsd:sequence>
         <xsd:attribute name="Name" type="xsd:string">
         </xsd:attribute>
         <xsd:attribute name="ID" type="xsd:string">
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

Figura 23. Il documento di esempio dello schema Schema1.xsd

L'ODA XML genera tre definizioni dell'oggetto di business per questo documento dello schema: `BOPrefix_TopLevel`, `BOPrefix_TopLevel_Customer` e `BOPrefix_TopLevel_TaxInfoType` (dove `BOPrefix` e `TopLevel` sono i valori di queste proprietà di configurazione ODA). Tutte e tre queste definizioni dell'oggetto di business hanno i seguenti elementi nelle relative informazioni specifiche sull'applicazione al livello di oggetto di business:

```
target_ns=http://www.ibm.com/ns1;elem_fd=unqualified;attr_fd=unqualified
```

Nota: Poiché l'elemento `schema` in Figura 23non include né l'attributo `elementFormDefault` né `attributeFormDefault`, questa informazione specifica sull'applicazione include le tag `elem_fd` e `attr_fd` impostate su `unqualified`. Per ulteriori informazioni, consultare "Nomi qualificati del componente" a pagina 70.

Un documento dello schema può definire solo uno spazio dei nomi di destinazione. Tuttavia, può includere elementi e attributi definiti in un altro spazio dei nomi di destinazione del documento dello schema utilizzando l'elemento `importa`.

Figura 24 mostra un documento dello schema che è basato sul documento `Schema1.xsd` definito in Figura 23 a pagina 67. Questo documento dello schema importa lo spazio dei nomi `ns2`, che dichiara il tipo complesso `TaxInfoType`, l'elemento `BillTo` e l'attributo `Name`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.example.com/ns2"
  xmlns:ns2="http://www.example.com/ns2">
<xsd:complexType name="TaxInfoType">
  <xsd:sequence>
    <xsd:element name="SSN" type="xsd:string"/>
    <xsd:element name="State" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:attribute name="Name" type="xsd:string"></xsd:attribute>
<xsd:complexType name="AddressType">
  <xsd:sequence>
    <xsd:element name="Zip" type="xsd:string">
    </xsd:element>
    <xsd:element name="Street" type="xsd:string">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="BillTo" type="ns2:AddressType">
</xsd:element>
</xsd:schema>
```

Figura 24. Importazione di uno spazio dei nomi di destinazione

La definizione di oggetto di business per un elemento root, `Customer2`, deve specificare questo spazio dei nomi alternativo per i suoi attributi che rappresentano i componenti TaxInfo, BillTo e Name XML, come descritto di seguito:

- L'attributo `TaxInfo` ha come relativo tipo una definizione di oggetto di business che rappresenta `TaxInfoType`, che è definito nello spazio dei nomi `ns2` (http://www.example.com/ns2) fare riferimento a Figura 25 a pagina 69.
- L'attributo `BillTo` ha la tag `elem_ns` nelle relative informazioni specifiche sull'applicazione per specificare lo spazio dei nomi `ns2` come la sorgente dell'elemento `BillTo` associato.
- L'attributo `Name` ha la tag `attr_ns` nelle relative informazioni specifiche sull'applicazione per specificare lo spazio dei nomi `ns2` come la sorgente del relativo attributo XML associato, `Name`.

Figura 25 mostra il documento dello schema che definisce i componenti XML `TaxInfoType`, `BillTo` e `Name` nello spazio dei nomi `ns2`.

```
<?xml version "1.0" encoding="UTF-8"?>
<schema smlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.example.com/ns2"
   attributeFormDefault="qualified" elementFormDefault="qualified"
   xmlns:ns2="http://www.example.com/ns2">
   <complexType name="TaxInfoType">
      <sequence>
         <element name="SSN" type="string">
         </element>
         <element name="State" type="string">
         </element>
      </sequence>
   </complexType>
   <attribute name="Name" type="string"
   </attribute>
   <complexType name="AddressType">
      <sequence>
         <element name="Zip" type="string">
         </element>
         <element name="Street" type="string">
         </element>
      </sequence>
   </complexType>
   <element name="BillTo" type="ns2:AddressType">
   </element>
</schema>
```

Figura 25. Definizione dello spazio dei nomi inferiore

Figura 25 mostra la definizione dell'oggetto di business per l'elemento root `Customer2`.

```
[BusinessObjectDefinition]
Name=TopLevel_Customer2
AppSpecificInfo=target_ns=http://www.example.com/ns1;elem_fd=qualified;
   attr_fd=qualified
...
   [Attribute]
   Name=Name
   Type=String
   AppSpecificInfo=attr_name=Name;type=attribute;attr_ns=http://www.example.com/ns2
   ...
   [End]
   [Attribute]
   Name=ID
   Type=String
   AppSpecificInfo=attr_name=ID;type=attribute
   ...
   [End]
   [Attribute]
   Name=schemaLocation
   Type=String
   AppSpecificInfo=attr_name=schemaLocation;type=xsischemalocation
   ...
   [End]
   [Attribute]
   Name=TaxInfo
   Type=TopLevel_TaxInfoType
   AppSpecificInfo=elem_name=TaxInfo
   ...
   [End]
   [Attribute]
   Name=BillTo
   Type=TopLevel_AddressType
   AppSpecificInfo=elem_name=BillTo;elem_ns=http://www.example.com/ns2
   ...
   [End]
...
[End]
```

Figura 26. Esempio di definizione dell'oggetto di business elemento root

Nomi qualificati del componente: In un documento XML, i nomi dei componenti associati con uno spazio dei nomi sono qualificati o non qualificati, come descritto di seguito:

- I nomi non qualificati *non* contengono un prefisso e *non* fanno parte di nessuno spazio di nomi.
- I nomi qualificati sono i seguenti:
  - Il nome componente contiene un prefisso associato con uno spazio dei nomi.

    e' possibile assegnare un prefisso ad uno o più spazi dei nomi. E' possibile dichiarare un prefisso dello spazio dei nomi con la tag `xmlns:`*`prefisso`*, dove *prefisso* è il prefisso dichiarato. In una definizione di un componente all'interno di un documento dello schema, questi nomi componenti sono *qualificati* perché hanno un prefisso preaggiunto al nome componente (*prefisso*:*componentName*).
  - I nomi non contengono un prefisso ma fanno parte dello spazio dei nomi predefinito (solo elementi).

    Il *spazio dei nomi predefinito* specifica lo spazio dei nomi da associare ai componenti che *non* includono un prefisso nei loro nomi componente. E' possibile dichiarare lo spazio dei nomi predefinito con la tag `xmlns`.

Perché il gestore dati XML possa gestire correttamente la conversione XML in oggetto di business, gli spazi dei nomi per il documento XML e per il documento dello schema devono corrispondere, come di seguito riportato:

- Se un documento dello schema specifica uno spazio dei nomi predefinito, il documento XML deve anche specificare lo spazio dei nomi predefinito.
- Se un documento dello schema *non* ha uno spazio dei nomi predefinito, il documento XML non può avere uno spazio dei nomi predefinito.

L'attributo `elementFormDefault` dell'elemento `schema` specifica se i nomi degli elementi dichiarati localmente sono qualificati. Per impostazione predefinita, gli elementi dichiarati localmente sono non qualificati ed appartengono allo spazio dei nomi predefinito. Il valore dell'attributo `elementFormDefault` determina il valore della tag `elem_fd` nelle informazioni specifiche sull'applicazione al livello dell'oggetto di business, come mostra Tabella 23.

Tabella 23. Impostazione della tag elem_fd

Valore di elementFormDefault	Valore della tag elem_fd
`non qualificato`(o l'attributo non è specificato per nulla)	`elem_fd=non qualificato`
`qualificato`	`elem_fd=qualificato`

Ad esempio, il documento dello schema in Figura 23 a pagina 67 *non* contiene l'attributo `elementFormDefault` nel suo elemento `schema`. Perciò, le informazioni specifiche sull'applicazione al livello dell'oggetto di business in *tutte* le definizioni dell'oggetto di business per questo documento dello schema (`BOPrefix_TopLevel`, `BOPrefix_TopLevel_Customer` e `BOPrefix_TopLevel_TaxInfoType`, dove `BOPrefix` e `TopLevel` sono i valori di queste proprietà di configurazione dell'ODA) contengono la tag:

```
elem_fd=non qualificato
```

Nota: Le informazioni specifiche sull'applicazione al livello dell'oggetto di business per queste tre definizioni dell'oggetto di business potrebbero contenere questa stessa tag se l'elemento schema `schema` del documento schema include l'attributo:

```
elementFormDefault=non qualificato
```

Se un singolo elemento XML include l'attributo `form`, questo valore dell'attributo `form` sovrappone qualsiasi impostazione dell'attributo `elementFormDefault`.

L'attributo `attributeFormDefault` dell'elemento `schema` specifica se i nomi degli elementi sono qualificati. Per impostazione predefinita, i nomi attributo non sono qualificati non appartengono a nessuno spazio dei nomi. Il valore dell'attributo `attributeFormDefault` determina il valore della tag `attr_fd` nelle informazioni specifiche sull'applicazione al livello dell'oggetto di business nello stesso modo in cui il valore dell'attributo `elementFormDefault` determina il valore della tag `elem_fd`, come mostra Tabella 24.

Tabella 24. Impostazione della tag attr_fd

Valore di attributeFormDefault	Valore della tag elem_fd
`non definito`(o l'attributo non è specificato per nulla)	`attr_fd=non qualificato`
`qualificato`	`attr_fd=qualificato`

Ad esempio, il documento dello schema in Figura 23 a pagina 67 *non* contiene l'attributo `attributeFormDefault` nel suo elemento `schema`. Perciò, le informazioni specifiche sull'applicazione al livello dell'oggetto di business in *tutte* le definizioni dell'oggetto di business per questo documento dello schema (`BOPrefix_TopLevel`, `BOPrefix_TopLevel_Customer` e `BOPrefix_TopLevel_TaxInfoType`, dove `BOPrefix` e `TopLevel` sono i valori di queste proprietà di configurazione dell'ODA) contengono la tag:

```
attr_fd=non qualificato
```

Nota: Le informazioni specifiche sull'applicazione al livello dell'oggetto di business per queste tre definizioni dell'oggetto di business potrebbero contenere questa stessa tag se l'elemento schema `schema` del documento schema include l'attributo:

```
attributeFormDefault=non qualificato
```

Informazioni specifiche sull'applicazione dell'attributo: L'attributo della definizione di un oggetto di business può rappresentare i seguenti componenti XML:

- "Per elementi XML" a pagina 73
- "Per elementi XML all'interno di un tipo complesso" a pagina 74
- "Per un attributo XML" a pagina 77
- "Per le istruzioni di elaborazione XML" a pagina 78
- "Per un commento XML" a pagina 78
- "Per un elemento o attributo XML che contenga caratteri speciali" a pagina 78
- "Per le posizioni dello schema XML" a pagina 79

Tabella 25 mostra le tag per le informazioni specifiche sull'applicazione al livello di attributo per questi differenti componenti XML insieme alle sezioni di questo manuale che descrivono queste tag con maggiori dettagli.

Tabella 25. Tag per le informazioni specifiche sull'applicazione attributo

Rappresentazione dell'attributo dell'oggetto di business	Informazioni specifiche sull'applicazione	Per ulteriori informazioni
Un elemento XML	elem_name=*nome dell'elemento XML* elem_ns=*spazio dei nomi per la definizione dell'elemento* elem_fd=*valore della* forma *attributo*	"Per elementi XML"
Un elemento XML all'interno di un tipo complesso	type=pcdata	"Per elementi XML all'interno di un tipo complesso" a pagina 74
Un attributo per un elemento XML	attr_name=*nome dell'attributo XML* type=attributo attr_ns=*spazio dei nomi per le definizioni dell'attributo* attr_fd=*valore di*form *attributo*	"Per un attributo XML" a pagina 77
Un elemento XML o un attributo il cui contenuto include caratteri speciali	escape=true	"Per un elemento o attributo XML che contenga caratteri speciali" a pagina 78
Un commento da aggiungere al documento XML	type=comment	"Per un commento XML" a pagina 78
Un'istruzione di elaborazione	type=pi	"Per le istruzioni di elaborazione XML" a pagina 78
L'attributo schemaLocation o noNamespaceSchemaLocationper un'istanza XML.	type=xsischemalocation type=xsinoNSlocation	"Per le posizioni dello schema XML" a pagina 79

Nota: Le informazioni specifiche sull'applicazione Attributo possono anche includere una tag con il formato (a | b | c) per specificare un attributo a cardinalità multipla che rappresenta una scelta ripetuta. Per ulteriori informazioni, consultare "Definizioni di un oggetto di business wrapper in base ai documenti dello schema" a pagina 61.

Per elementi XML: Se un attributo dell'oggetto di business rappresenta un elemento XML, le relative informazioni specifiche sull'applicazione devono includere la tag elem_name per identificare l'elemento associato:

elem_name=*nome dell'elemento XML*

I nomi elemento XML può contenere caratteri speciali (ad esempio punti e trattini). Tuttavia, i nomi degli attributi dell'oggetto di business non possono contenere questi caratteri speciali. Perciò, il nome dell'elemento XML deve essere specificato nella tag elem_name. Per denominare l'attributo dell'oggetto di business, l'ODA XML rimuove qualsiasi carattere speciale nel nome dell'elemento XML.

Un attributo dell'oggetto di business può rappresentare un elemento XML in uno dei seguenti casi:

- Se l'elemento XML è (o contiene) un XML tipi complessi

  In questo caso, l'attributo dell'oggetto di business è un *complesso* attributo il cui tipo di dati è la definizione dell'oggetto di business che rappresenta il tipo

complesso XML. La relativa tag `elem_name` (nelle informazioni specifiche sull'applicazione nell'attributo) contiene il nome dell'elemento XML (o tipo complesso).

- Se l'elemento XML fa parte di un tipo complesso XML

  In questo caso, l'attributo dell'oggetto di business è un attributo *semplice* del tipo `String`. Le relative informazioni specifiche sull'applicazione includono la tag `elem_tag` (che contiene il nome dell'elemento XML all'interno del tipo complesso) e la tag `type=pcdata`. Per ulteriori informazioni, consultare "Per elementi XML all'interno di un tipo complesso".

Un attributo dell'oggetto di business che rappresenta un elemento XML può anche includere le seguenti tag nelle relative informazioni specifiche sull'applicazione:

- La tag `elem_fd` specifica l'impostazione dell'attributo XML attributo`forma`, che indica se i nomi degli elementi dichiarati localmente sono qualificati o non qualificati. Se un elemento XML ha specificato l'attributo `form=qualificato`, il valore di elem_fd è impostato sul valore dell'attributo form.

  Ad esempio, supporre che un elemento XML dichiarato localmente ha la seguente definizione:

  ```
  <xsd:element ref="Name" form="qualified"></xsd:element>
  ```

  Il relativo attributo dell'oggetto di business associato dovrebbe avere il seguente formato:

  ```
  [Attribute]
  Name=Name
  Type=String
  AppSpecificInfo=elem_name=Name;elem_fd=qualified;
  ...
  ```

  Se un elemento XML *non* specifica l'attributo `form`, il valore dell'attributo `elementFormDefault` (sull'elemento `schema`) determina se i nomi dell'elemento sono qualificati. Per ulteriori informazioni, consultare "Nomi qualificati del componente" a pagina 70.

- La tag `elem_ns` specifica lo spazio dei nomi di destinazione per l'elemento XML, se tale spazio dei nomi è *differente* dallo spazio dei nomi di destinazione del documento dello schema. Questa tag è richiesta quando un documento dello schema utilizza spazi dei nomi multipli. Se l'elemento XML riferito in un documento dello schema è definito nello spazio del nome di destinazione di *qualche altro documento dello schema*, questa tag elenca il nome di quello spazio del nome.

  Ad esempio, supporre che un elemento XML all'interno di un tipo complesso ha la seguente definizione:

  ```
  <xsd:element ref="ns2:BillTo"></xsd:element>
  ```

  Il relativo attributo dell'oggetto di business associato dovrebbe avere il seguente formato:

  ```
  [Attribute]
  Name=BillTo
  Type=String
  AppSpecificInfo=elem_name=BillTo;elem_ns=http:/www.imb.com/ns2;
  ...
  ```

  Per un esempio dettagliato di un documento dello schema che include elementi XML definiti in un secondo spazio del nome, fare riferimento a "Spazio dei nomi dello schema" a pagina 65.

Per elementi XML all'interno di un tipo complesso: Se una definizione dell'oggetto di business rappresenta un tipo complesso XML (`complexType`), i contenuti di questo tipo complesso sono rappresentati da attributi semplici (`String`) all'interno di

questa definizione dell'oggetto di business. Queste informazioni specifiche sull'applicazione degli attributi dell'oggetto di business *devono* includere la tag `elem_name` per identificare il nome dell'elemento.

Nota: Per ulteriori informazioni su questa tag, fare riferimento a "Per elementi XML" a pagina 73.

All'interno di un tipo complesso XML, un attributo dell'oggetto di business può rappresentare i generi del contenuto del tipo complesso mostrati in Tabella 26.

Tabella 26. Contenuti di un tipo complesso XML

Tipo dell'elemento del tipo complesso	Descrizione	Informazioni specifiche sull'applicazione dell'attributo
Elemento semplice XML	Un elemento che contiene *solo* contenuto di caratteri. Se non può contenere altri elementi o attributi XML. Si può verificare solo all'interno di un tipo complesso	`type=pcdata`
Contenuto semplice	Solo dati di caratteri (nessun elemento)	`type=pcdata;notag`
Elemento XML element con attributi	Un elemento che contiene *sia* dati di carattere *che* alcune combinazioni di sottoelementi e attributi	Nessuno. Questo tipo di elemento XML deve essere rappresentato con una definizione dell'oggetto di business, non come un singolo attributo dell'oggetto di business. Per ulteriori informazioni, consultare "Definizioni dell'oggetto di business regolare nei documenti dello schema" a pagina 59.

Come esempio dell'utilizzo della tag `type=pcdata`, il documento dello schema in Figura 23 a pagina 67 contiene una definizione per il tipo complesso `TaxInfoType`, che contiene solo elementi XML semplici. La definizione dell'oggetto di business per il tipo complesso XML TaxInfoType (`BOPrefix_TopLevel_TaxInfoType`, dove `BOPrefix` e `TopLevel` sono i valori delle proprietà di configurazione ODA rispettivamente chiamate) dovrebbe contenere due attributi. Ogni attributo dovrebbe avere il nome dell'elemento XML associato nelle sue informazioni specifiche sull'applicazione, come mostra Figura 27. Poiché questo tipo complesso contiene solo elementi XML semplici (nessun sottoelementi o attributi), i corrispondenti attributi dell'oggetto di business devono anche contenere la tag `type=pcdata` nelle loro informazioni specifiche sull'applicazione.

```
[BusinessObjectDefinition]
Name=BOPrefix_TopLevel_TaxInfoType
AppSpecificInfo=target_ns=;elem_fd=unqualified;attr_fd=unqualified
...
[End]

[Attribute]
Name=SSN
Type=String
AppSpecificInfo=elem_name=SSN;type=pcdata
...
[End]

[Attribute]
Name=State
Type=String
AppSpecificInfo=elem_name=SSN;type=pcdata
...
[End]
```

Figura 27. Esempio di definizione dell'oggetto di business per un tipo complesso XML con elementi semplici

Come esempio della parola chiave `notag` utilizzata in congiunzione con la tag `type=pcdata`, supporre che un elemento XML denominato `Price` sia il tipo complesso `PriceType`, che abbia solo contenuto semplice; ossia, contenga solo dati di carattere. In questo caso, l'elemento `simpleContent` definisce un attributo denominato `Currency` e richiede dati per `Price`:

```
<xsd:element name="Price" type="PriceType">
    <xsd:complexType name="PriceType">
        <xsd:simpleContent>
            <xsd:extension base="xsd:decimal">
                <xsd:attribute name="Currency" type="xsd:NMTOKEN"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</element>
```

La definizione dell'oggetto di business per il tipo complesso `PriceType` contiene un attributo per contenere i dati carattere associati con contenuto semplice. Le informazioni specifiche sull'applicazione per questo attributo devono contenere le seguenti definizioni:

```
type=pcdata;notag
```

La parola chiave `notag` evita che il gestore dati XML generi tag di avvio duplicate (una per la definizione dell'oggetto di business e una per l'attributo). Il gestore dati XML crea una tag XML di avvio e di fine per ogni attributo dell'oggetto di business *a meno che* `notag` non appaia nelle informazioni specifiche sull'applicazione per questo attributo.

La definizione `Price` dell'oggetto di business secondaria dovrebbe essere simile a:

```
[BusinessObjectDefinition]
Name = Price
AppSpecificInfo =

[Attribute]
Name = Currency
Type = String
AppSpecificInfo = attr_name=Currency;type=attribute;
...
[End]

[Attribute]
Name = Price
```

```
Type = String
AppSpecificInfo = elem_name=Price;type=pcdata;notag
...
[End]
```

Un attributo dell'oggetto di business object deve esistere per poter contenere i dati `Price`. L'attributo per i dati `Price` deve specificare `notag` nelle relative informazioni specifiche sull'applicazione per impedire al gestore dati di creare una tag di avvio e di fine per questo attributo. In questo caso, il gestore dati *non* genera un nuovo elemento XML per i dati `Price` ma aggiunge semplicemente i dati all'elemento principale.

Inoltre, la definizione dell'oggetto di business deve contenere un altro attributo che deve esistere in modo da avere il valore dell'attributo `Currency`. Se il contenuto semplice ha attributi, la definizione dell'oggetto di business object dovrà avere anche un attributo per ogni attributo XML. Le informazioni specifiche sull'applicazione dell'attributo devono includere la tag `type=attribute`. Per ulteriori informazioni, consultare "Per un attributo XML".

Per un attributo XML: Quando una definizione dell'oggetto di business rappresenta un elemento o un tipo complesso, qualsiasi attributo che il documento dello schema dichiara per questo elemento viene rappresentato come un attributo all'interno della definizione di oggetto di business. Quando un attributo dell'oggetto di business rappresenta un attributo di un elemento XML, le relative informazioni specifiche sull'applicazione *devono* includere le seguenti tag:

- Il tag`attr_name`:

  `attr_name=`*`nome dell'attributo XML`*
- La tag `type`:

  `type=attribute`

Nota: Per ulteriori informazioni sull'utilizzo di queste tag, fare riferimento a "Per un attributo XML" a pagina 50. Per un esempio sull'utilizzo della tag `type=attribute`, fare riferimento a "Per elementi XML all'interno di un tipo complesso" a pagina 74.

Un attributo dell'oggetto di business che rappresenti un attributo XML può includere anche le seguenti tag nelle relative informazioni specifiche sull'applicazione:

- La tag `attr_fd` specifica l'impostazione dell'attributo `form` dell'attributo XML, che indica se i nomi dell'attributo sono qualificati o non qualificati. Se un attributo XML ha l'attributo `form="qualified"` per `form="unqualified"` specificato, attr_fd ha un valore specificato nell'attributo form.

  Ad esempio, supporre che un attributo XML abbia la seguente definizione:

  ```
  <xsd:attribute ref="Name" form="qualified"></xsd:attribute>
  ```

  Il relativo attributo dell'oggetto di business associato dovrebbe avere il seguente formato:

  ```
  [Attribute]
  Name=Name
  Type=String
  AppSpecificInfo=attr_name=Name;type=attribute;attr_fd=qualified
  ...
  ```

  Se un attributo XML non specifica l'attributo `form`, il valore dell'attributo `attributeFormDefault` (sull'elemento `schema`) determina se i nomi dell'attributo sono qualificati. Per ulteriori informazioni, consultare "Nomi qualificati del componente" a pagina 70.

- La tag `attr_ns` specifica lo spazio del nome di destinazione per l'attributo XML, se tale spazio del nome è *differente* dallo spazio del nome di destinazione del documento dello schema. Questa tag è richiesta quando un documento dello schema utilizza spazi dei nomi multipli. Se l'attributo XML riferito in un documento dello schema è definito nello spazio dei nomi di destinazione di *qualche altro documento dello schema*, questa tag elenca il nome di quello spazio del nome.

  Ad esempio, supporre che un attributo XML abbia la seguente definizione:

  ```
  <xsd:attribute ref="ns2:Name"></xsd:attribute>
  ```

  Il relativo attributo dell'oggetto di business associato dovrebbe avere il seguente formato:

  ```
  [Attribute]
  Name=Name
  Type=String
  AppSpecificInfo=attr_name=Name;attr_ns=http://www.example.com/ns2;
     type=attribute
  ...
  ```

  Per un esempio dettagliato di un documento dello schema che includa attributi XML definiti in un secondo spazio del nome, fare riferimento a "Spazio dei nomi dello schema" a pagina 65.

Per un elemento o attributo XML che contenga caratteri speciali: Gli attributi dell'oggetto di business che rappresentano elementi XML o attribtuti XML che includono caratteri speciali nel loro contenuto richiedono l'elaborazione escape da parte del gestore dati XML. Per avvisare il gestore dati della necessità di effettuare l'elaborazione escape, le informazioni specifiche sull'applicazione dell'attribute dell'oggetto di business devono contenere la seguente tag:

```
escape=true
```

I passaggi per specificare l'elaborazione escape per un documento XML che utilizza un documento dello schema per descrivere i suoi schemi sono gli stessi utilizzati per specificare l'elaborazione escape per un documento XML che utilizza un DTD per descrivere i suoi schemi. Per ulteriori informazioni, consultare "Per un elemento o attributo XML che contenga caratteri speciali" a pagina 52.

Per un commento XML: Quando il gestore dati XML converte un oggetto di business in un documento XML, è possibile specificare che il gestore dati aggiunge i commenti XML al documento XML includendo la seguente tag nelle informazioni specifiche sull'applicazione di un attributo:

```
type=comment
```

ODA XML *non* genera automaticamente attributi dell'oggetto di business per i commenti XML. E' necessario aggiungere manualmente tali attributi. I passaggi per aggiungere commenti ad un documento XML descritto da un documento dello schema sono gli stessi utilizzati per definire un commento per un documento XML che è descritto da un DTD. Per ulteriori informazioni, consultare "Per un commento XML" a pagina 53.

Per le istruzioni di elaborazione XML: Se il documento XML contiene istruzioni di elaborazione XML, alcune definizioni dell'oggetto di business associate con il documento dello schema devono contenere un attributo per contenere il valore dell'elaborazione. Le informazioni specifiche sull'applicazione di questo attributo devono contenere la seguente tag `type`:

```
type=pi
```

Ad esempio, quando il gestore dati XML converte un documento XML in un oggetto di business, colloca la versione XML in un attributo speciale della definizione dell'oggetto di business di livello superiore chiamato `XMLDeclaration`. L'oggetto di business di livello superiore in Figura 19 a pagina 58 mostra l'attributo XMLDeclaration all'interno della definizione di oggetto di business TopLevel_Customer. Per ulteriori informazioni riguardo alla tag `type=pi`, fare riferimento a "Per istruzioni di elaborazione XML" a pagina 54.

Per le posizioni dello schema XML: I documenti XML che fanno riferimento alle posizioni dello schema per i loro documenti dello schema devono includere le seguenti informazioni:

- Dichiarazione dello Spazio del nome dell'istanza dello schema XML e un'associazione del prefisso `xsi` per questo spazio del nome
- Inclusione di uno dei seguenti attributi istanza dello schema XML:
  - `xsi:schemaLocation`

    Questo attributo associa il nome di uno spazio del nome con una posizione dello schema. Se il documento dello schema utilizza uno spazio del nome di destinazione, il nome di tale spazio del nome di destinazione deve corrispondere allo spazio del nome che l'attributo `xsi:schemaLocation` (nel documento XML) specifica.
  - `xsi:noNamespaceSchemaLocation`

    Questo attributo identifica una posizione dello schema. Se il documento dello schema *non* utilizza uno spazio del nome di destinazione, il documento XML include l'attributo `noNamespaceSchemaLocation` per identificare una singola posizione dello schema.

Ad esempio, supporre che il documento XML abbia la dichiarazione dello spazio del nome in Figura 28.

```
<order xmlns="http://sampleDoc.org.ord"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.example.com/order order.xsd">
   ...
/<order>
```

Figura 28. Esempio di definizione di una posizione dello schema in un documento XML

Il documento dello schema dovrebbe avere la seguente dichiarazione dello spazio del nome, che definisce uno spazio del nome di destinazione:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns="http://www.example.com/order"
     targetNamespace="http://www.example.com/order">
```

Affinché l'oggetto di business che rappresenta questo documento XML sia in grado di contenere le informazioni nell'attributo `schemaLocation`, le relative definizioni dell'oggetto di business devono fornire un attributo per tali informazioni della posizione dello schema. Nell'elemento root della definizione di un oggetto di business, questa informazioni della posizione dello schema vengono rappresentate come segue:

- Per un documento XML che utilizza l'attributo `schemaLocation`, la definizione dell'oggetto di business di livello superiore deve contenere un attributo con le seguenti proprietà:
  - Il nome dell'attributo è `schemaLocation` e tale attributo è del tipo `String`.
  - L'attributo deve avere la seguente tag `type` nelle relative informazioni specifiche sull'applicazione:

type=xsischemalocation

  - Il valore di questo attributo dell'oggetto di business è il valore dell'attributo schemaLocation.
- Per un documento XML che utilizza l'attributo noNamespaceSchemaLocation, la definizione dell'oggetto di business dell'elemento root contiene un attributo con le seguenti proprietà:
  - Il nome dell'attributo è noNamespaceSchemaLocation e tale attributo è del tipo String.
  - L'attributo deve avere la seguente tag type nelle relative informazioni specifiche sull'applicazione:

    type=xsinoNSlocation

La definizione dell'oggetto di business elemento root per il documento dello schema che rappresenta tale documento XML include i seguenti attributi per la posizione dello schema:

```
[Attribute]
Name=schemaLocation
Type=String
AppSpecificInfo=type=xsischemalocation;
```

Nota: Quando un attributo rappresenta l'attributo XML schemaLocation o noNamespaceSchemaLocation, *non* richiede la tag type=attribute nelle relative informazioni specifiche sull'applicazione.

L'ODA XML determina se generare un attributo schemaLocation o noNamespaceSchemaLocation nell'oggetto di business di livello superiore basato sul valore della relativa proprietà di configurazione ODA DoctypeorSchemaLocation.

Creazione delle definizioni di business da documenti dello schema

Un documento dello schema descrive e limita il contenuto di un oggetto XML. Perciò, il documento dello schema è molto utile per ottenere le informazioni necessarie per la definizione dell'oggetto di business. Per tradurre le informazioni della struttura nel documento dello schema per una definizione dell'oggetto di business, è possibile utilizzare l'ODA XML (Object Discovery Agent). Per ulteriori informazioni su ODA XML, fare riferimento a "Utilizzo di ODA XML per creare definizioni degli oggetti di business" a pagina 83.

Strutture del documento dello schema supportate

L'ODA XML associa la dichiarazione di un elemento nello schema ad un attributo nella definizione dell'oggetto di business. Il tipo di attributo dell'oggetto di business dipende dal tipo specificato nella dichiarazione dell'elemento XML. Un tipo semplice si associa ad un tipo Java String. I tipi complessi si associano ad una definizione dell'oggetto di business.

L'ODA XML supporta le strutture del documento dello schema inclusi:

- I tipi semplici — I tipi semplici possono essere nucleari (incorporati o derivati utilizzando limitazioni), di elenco o di unione. L'ODA XML associa tutti questi tipi al tipo di stringa dell'attributo dell'oggetto di business. Per ulteriori informazioni, consultare "Per elementi XML all'interno di un tipo complesso" a pagina 74.
- Il carattere jolly dell'elementoany — L'ODA XML associa il carattere jolly any ad un attributo semplice nella definizione dell'oggetto di business. Durante

l'esecuzione, è necessario che gli utenti configurino tale attributo (aggiungendo le appropriate informazioni specifiche sull'applicazione) in base alla conoscenza dei dati.

- `anyAttribute` elemento — Questo elemento abilita l'utente ad estendere il documento XML con attributi che non sono specificati nello schema. L'ODA XML richiede all'utente un nome per questo attributo e quindi associa l'attributo ad un attributo semplice nella definizione dell'oggetto di business (fare riferimento a Figura 50 a pagina 229). L'ODA ignora qualsiasi attributo `namespace` e `processContents` per il carattere jolly `anyAttribute`.
- `gruppi` sequence — Questo modello di gruppo richiede elementi secondari per apparire in un ordine specifico. L'ODA XML associa gli elementi secondari in un gruppo `sequence` per gli attributi all'interno della definizione dell'oggetto di business. Inoltre, determina il tipo di questi attributi dall'attributo XML `type`.
- `gruppi di` choice — Questo modello di gruppo indica che solo una delle dichiarazioni corrispondenti dell'elemento conforme deve apparire in una istanza. L'ODA XML associa gli elementi secondari in un gruppo `choice` per gli attributi all'interno di una definizione dell'oggetto di business wrapper. Una definizione dell'oggetto di business principale contiene un singolo attributo a cardinalità multipla il cui tipo è la definizione dell'oggetto di business wrapper. Per ulteriori informazioni, consultare "Definizioni di un oggetto di business wrapper in base ai documenti dello schema" a pagina 61.
- `gruppi` all — L'ODA XML associa gli elementi secondari in un gruppo `all` per gli attributi all'interno di una definizione dell'oggetto di business wrapper. Una definizione dell'oggetto di business principale contiene un singolo attributo a cardinalità multipla il cui tipo è la definizione dell'oggetto di business wrapper. Per ulteriori informazioni, consultare "Definizioni di un oggetto di business wrapper in base ai documenti dello schema" a pagina 61.
- `elemento` include — L'effetto dell'elemento `include` è di portare in un documento dello schema tutte le definizioni e le dichiarazioni contenute nel documento dello schema incluso. L'ODA XML richiede che si fornisca l'intero percorso del documento dello schema che deve essere incluso. Questo percorso è l'attuale posizione del documento dello schema incluso nel file system, *non* la sua URI. Se si specifica la posizione del documento dello schema con la proprietà ODA XML `FileName`, tutti i documenti dello schema inclusi devono esistere nella stessa posizione.

  Nota: Quando si include un documento dello schema, assicurarsi che il documento dello schema incluso *non* dichiari elementi globali che abbiano gli stessi nomi del documento dello schema che esegue `include`.
- `elemento` restriction — Questo elemento consente di far derivare tipi complessi per limitazione; ossia, di limitare un tipo complesso rimuovendo o limitando i suoi attributi o il contenuto. L'ODA XML supporta l'elemento limitazione nei seguenti contesti:
  - Per limitazioni di contenuto semplice, ODA XML non effettua alcuna associazione per le limitazioni definite per il contenuto semplice e che coinvolge aspetti del tipo derivato. ODA XML elabora le limitazioni di qualsiasi attributo definito nel tipo derivato.
  - Per limitazioni di contenuto complesso, ODA XML supporta l'eliminazione o la limitazione degli attributi e modello di contenuto del tipo complesso base, come di seguito riportato:
    - Per un modello di contenuto limitato, ODA crea una definizione dell'oggetto di business *sia per* il tipo complesso base che per il tipo derivato. Tuttavia, l'ODA XML presume che la sintassi dello schema sia

valida e segua le linee guida specifiche per le limitazioni dello schema W3C; *non* effettua convalide se i tipi derivati sono limitazioni valide del tipo base.

- Per le limitazioni di un attributo, ODA XML presume che tutti gli attributi del tipo base siano trasmessi al tipo derivato. Perciò, la definizione dell'oggetto di business presenta attributi per tutti gli attributi XML definiti nella limitazione così come qualsiasi altro attributo definito nel tipo base (che non sono inclusi nella limitazione).

- `import` elemento — Questo elemento consente ad un documento XML di riferirsi a componenti di un altro spazio del nome. Specificando le tag `elem_ns` o `attr_ns` nelle informazioni specifiche dell'applicazione dell'attributo, la definizione dell'oggetto di business può identificare uno spazio del nome importato. Per ulteriori informazioni, consultare "Spazio dei nomi dello schema" a pagina 65.
- Derivazione tipi complessi — L'ODA XML supporta tipi complessi derivati utilizzando limitazioni ed estensioni. Per questi costrutti, ODA crea una definizione dell'oggetto di business per il tipo complesso derivato in base al relativo modello di contenuto derivato.
- Sostituzione del tipo — La proprietà ODA XML `TypeSubstitution` fornisce supporto per la sostituzione del tipo in base all'attributo xsi:type nello schema. Quando l'utente imposta `TypeSubstitution` su Vero nel menu in un menu a discesa nell'ODA, ODA XML genera un oggetto wrapper addizionale con l'ASI "typeSub=true" e gli oggetti di business che rappresentano il tipo base e i relativi tipi derivati come tipi secondari di questo oggetto wrapper. Per ulteriori informazioni, consultare "Sostituzione del tipo nelle definizioni dell'oggetto di business in base ai documenti dello schema" a pagina 62.

Strutture del documento dello schema non supportate

ODA XML è in grado di elaborare la maggior parte di documenti dello schema. Tuttavia, *non* supporta le seguenti strutture del documento dello schema:

- Sostituzione di un elemento utilizzando substitutionGroups— Il gestore dati XML *non* supporta la sostituzione di un elemento durante l'esecuzione perché l'ODA XML non supporta i gruppi di sostituzione negli schemi XML.
- Ridefinizione di tipi e gruppi — L'elemento `redefine` consente di ridefinire un tipo (semplice o complesso) o un gruppo (elemento o attributo) in modo che solo certi comportamenti siano supportati.
- Tipi o elementi senza contenuto. In questo caso, l'ODA XML genererà un oggetto di business senza attributi. Gli utenti possono risolvere questo problema aggiungendo un attributo fittizio alla definizione dell'oggetto di business e controllando il relativo indicatore "Chiave". Per ulteriori informazioni riguardo agli attributi chiave, fare riferimento a "Proprietà degli attributi Chiave e Chiave esterna" a pagina 35.

Creazione delle definizioni di un oggetto di business

Un documento XML document può avere o un DTD o un documento dello schema per definire la sua struttura. Le definizioni dell'oggetto di business che rappresentano gli elementi in un documento XML devono contenere le informazioni sulla struttura del documento. Per creare oggetti di business che devono essere elaborati dal gestore dati XML, il gestore dati XML deve essere in grado di localizzare le definizioni dell'oggetto di business che contengono le informazioni strutturali per ogni documento XML document che deve essere elaborato. E' possibile generare definizioni dell'oggetto di business per i documenti XML in uno dei seguenti modi:

- "Utilizzo di ODA XML per creare definizioni degli oggetti di business"
- "Creazione manuale delle definizioni dell'oggetto di business"

Entrambe queste tecniche implicano l'utilizzo dello strumento Business Object Designer Express. Questa sezione fornisce una panoramica su come utilizzare Business Object Designer Express per generare le definizioni dell'oggetto di business per i documenti XML. Per una descrizione esauriente di Business Object Designer Express , fare riferimento alla *Guida allo sviluppo di Business Object*.

Nota: Alcuni connettori che utilizzano il gestore dati XML richiedono un oggetto di business wrapper di livello superiore che abbia gli oggetti di business del contenuto come elementi secondari. In base alla struttura dell'oggetto di business richiesta dal connettore che si utilizza, perciò, potrebbe essere necessario aggiungere oggetti di business generati come secondari da un altro oggetto di business. Fare riferimento alla documentazione per i connettori per ulteriori informazioni sulla struttura dei relativi oggetti di business.

Utilizzo di ODA XML per creare definizioni degli oggetti di business

L'ODA XML (Object Discovery Agent) crea definizioni degli oggetti di business per un documento XML in base o ai suoi DTD o al documento dello schema. L'ODA esamina il DTD o il documento dello schema per ottenere informazioni sulla struttura del documento XML. Quindi l'ODA scrive le definizioni dell'oggetto di business in un file che può essere caricato nel sistema di integrazione business.

Nota: Se un documento XML *non* ha un DTD o un documento dello schema, è possibile creare manualmente una definizione dell'oggetto di business per il documento. Per ulteriori informazioni, consultare "Creazione manuale delle definizioni dell'oggetto di business".

L'ODA XML crea le definizioni dell'oggetto di business che si adattano ai requisiti del gestore dati XML. L'ODA aggiunge l'attributo richiesto `ObjectEventId` a tutte le definizioni dell'oggetto di business. Inoltre, aggiunge il numero della versione repository all'inizio del file dell'oggetto di business se viene specificato, che è necessario per importare un definizione dell'oggetto di business nel sistema di integrazione business InterChange Server Express. Queste definizioni dell'oggetto di business *non* richiedono modifiche aggiuntive. Tuttavia, qualora fosse necessario effettuare modifiche, fare riferimento a "Modifica delle informazioni nella definizione di un oggetto di business" a pagina 230.

Per ulteriori informazioni su come utilizzare ODA XML, fare riferimento a "Utilizzare l'ODA XML", a pagina 217. Questa appendice descrive come installare e configurare ODA XML. Inoltre descrive come utilizzerà l'ODA XML in Business Object Designer Express per generare una definizione dell'oggetto di business. Per informazioni su come eseguire Business Object Designer Express , fare riferimento alla *Guida allo sviluppo di Business Object*.

Creazione manuale delle definizioni dell'oggetto di business

Questa sezione descrive come creare manualmente le definizioni dell'oggetto di business per rappresentare documenti XML. Assicurarsi che siano state definite correttamente le definizioni dell'oggetto di business, attributi inclusi e le informazioni specifiche sull'applicazione.

Nota: Se un documento XML *non* ha un DTD o un documento dello schema, sarà necessario creare manualmente una definizione dell'oggetto di business per il documento. Se invece, esistono un DTD o un documento dello schema, IBM raccomanda l'utilizzo di ODA XML per creare una definizione dell'oggetto di business.

La descrizione del formato di un documento XML per un DTD o un documento dello schema descrive le definizioni dell'oggetto di business che crea ODA XML. Tabella 9 a pagina 29 mostra le sezioni di questo manuale che descrivono il formato dei documenti XML che hanno un modello di dati corrispondente per descrivere il loro schema. Come descritto in queste sezioni, le definizioni dell'oggetto di business si adattano ai requisiti del gestore dati XML. Perciò, è possibile seguire queste descrizioni quando sarà necessario creare le definizioni dell'oggetto di business manualmente.

Nei seguenti passaggi, *ElementTypeName* è il tipo dell'elemento XML rappresentato dal costrutto dell'oggetto di business (attributo o oggetto di business). Per definire un oggetto di business in base ad un documento XML:

1. Creare la definizione dell'oggetto di business di livello superiore. Il nome di questa definizione dell'oggetto di business dovrebbe essere l'elemento di livello più altro nel documento XML (nome del DTD o documento dello schema, ad esempio) nel formato: *BOPrefix_TopLevelName*.

   Nota: Alcuni connettori richiedono un oggetto di business wrapper che abbia l'oggetto di business del contenuto come elemento secondario. Fare riferimento a "Definizioni dell'oggetto di business wrapper in base ai DTD" a pagina 46 per ulteriori informazioni.
2. Nella definizione dell'oggetto di business di livello superiore, creare attributi per gli elementi XML.

   Per una definizione dell'oggetto di business basata o su un DTD su un documento dello schema, sono richiesti i seguenti attributi:

   - `XMLDeclaration` – Le informazioni specifiche sull'applicazione dell'attributo sono `type=pi`.
   - Un attributo per rappresentare l'elemento root nel DTD o documento dello schema – L'attributo contiene un oggetto di business secondario a cardinalità singola e le relative informazioni specifiche sull'applicazione stabiliscono il nome dell'elemento con la tag `elem_name`.

   Per informazioni generali su questi attributi richiesti, fare riferimento a "Struttura oggetto di business" a pagina 33. Inoltre, questo manuale fornisce le seguenti informazioni sulla struttura della definizione dell'oggetto di business in base ai DTD e ai documenti dello schema:

Modello di dati	Per ulteriori informazioni
DTD(Document type definition)	"Struttura dell'oggetto di business per DTD" a pagina 42
Documento dello schema	"Definizioni dell'oggetto di business richieste per i documenti dello schema" a pagina 57

3. Creazione della definizione dell'oggetto di business dell'elemento root, che è un oggetto secondario della definizione dell'oggetto di business di livello superiore. Questo oggetto contiene gli attributi per l'elemento root XML. Il nome di questa definizione dell'oggetto di business dovrebbe essere l'elemento root nel documento XML nel formato: *BOPrefix_RootElementName_TopLevelName*

4. Nella definizione dell'oggetto di business dell'elemento root, creare un attributo dell'oggetto di business per ogni elemento contenuto. Tenere presente quanto segue:
   - Il nome dell'attributo dell'oggetto di business name non deve essere uguale al nome dell'elemento (o attributo) XML. Le informazioni specifiche sull'applicazione sono utilizzate per specificare il nome dell'elemento (o attributo).
   - Gli attributi XML devono essere i primi attributi nella definizione dell'oggetto di business.
   - Determinazione del tipo:
     – Un tipo stringa è un elemento a cardinalità-1 con nessun elemento del contenuto o dichiarazione dell'elenco attributo associata.
     – Un tipo dell'oggetto di business è un elemento contenuto a cardinalità-n, o un elemento contenuto con un elemento del contento o specifica (o specifiche) dell'attributo associata.
   - Le informazioni specifiche sull'applicazione sono richieste per un attributo di tipo stringa, un elemento di tipo misto o un elenco di scelta a cardinalità n.

     Per informazioni generali su queste informazioni specifiche sull'applicazione, fare riferimento a "informazioni specifiche sull'applicazione" a pagina 37. Inoltre, questo manuale fornisce le seguenti informazioni sulle applicazioni specifiche degli attributi dell'oggetto di business in base ai DTD e ai documenti dello schema:

Modello di dati	Per ulteriori informazioni
DTD(Document type definition)	"Informazioni specifiche sull'applicazione per componenti XML nei DTD" a pagina 45
Documento dello schema	"Informazioni specifiche sull'applicazione per componenti XML nei documenti dello schema" a pagina 65

   Nota: Per un documento XML con una definizione dello schema, la definizione dell'oggetto di business dell'elemento root potrebbe richiedere anche un attributo per la posizione dello schema location. Per ulteriori informazioni, consultare "Definizioni dell'oggetto di business richieste per i documenti dello schema" a pagina 57.
5. Creare le definizioni dell'oggetti di business secondario per tutti gli elementi contenuti. Utilizzare le regole elencate in precedenza.

Conversione degli oggetti di business in documenti XML

Per convertire un oggetto di business in un documento XML, il gestore dati XML esegue un loop attraverso gli attributi nella definizione dell'oggetto di business in ordine sequenziale. Genera ripetitivamente XML in base all'ordine in cui gli attributi appaiono nell'oggetto di business e nei suoi elementi secondari.

Il gestore dati XML elabora gli oggetti di business in un documento XML nel modo seguente:

1. Il gestore dati crea un documento per contenere i dati XML.
2. Il gestore dati esamina le informazioni specifiche sull'applicazione nel livello superiore della definizione dell'oggetto di business per determinare se ci siano meta oggetti secondari (quelli i cui nomi sono elencati nella tag `cw_mo_`*`label`*

delle informazioni specifiche sull'applicazione al livello dell'oggetto di business). Il gestore dati *non* include questi attributi nel documento XML.

3. Il gestore dati esegue un loop attraverso i rimanenti attributi della definizione dell'oggetto di business.

   Il gestore dati genera XML per ogni attributo utilizzando le seguenti regole:

   - Crea una tag di avvio e di fine per ogni tipo semplice di attributo `String` finché la stringa `notag` non appare nelle informazioni specifiche sull'applicazione per quell'attributo. Se la tag di avvio è aperta e l'attributo dell'oggetto di business che è stato elaborato *non* rappresenta un attributo XML, il gestore dati XML chiude la tag di avvio (ossia, aggiunge il carattere ">" al documento XML).
   - Se l'attributo rappresenta un oggetto di business, il gestore dati XML apre una tag di avvio, effettua una richiesta ricorrente per recuperare gli attributi nell'oggetto di business secondario, genera XML per gli attributi dell'oggetto di business secondario e quindi genera una tag di fine per l'elemento. Il gestore dati XML utilizza le informazioni specifiche sull'applicazione *al livello dell'attributo* `elem_name` come il nome dell'elemento. Per attributi a cardinalità multipla, questo processo è ripetuto per ogni istanza dell'oggetto di business nell'array.
   - Se l'attributo contiene le informazioni specifiche sull'applicazione `xsiType`, il gestore dati XML scrive per esteso un attributo per l'elemento corrente del formato `xsi:type=ValueofXsiType`. Ad esempio, se l'attributo dell'oggetto di business specifica `xsiType=ShirtType` allora il corrispondente attributo XML sarà `xsi:type="ShirtType"`. Per ulteriori informazioni riguardo alla sostituzione del tipo, fare riferimento a "Sostituzione del tipo nelle definizioni dell'oggetto di business in base ai documenti dello schema" a pagina 62.
   - Per gli attributi che rappresentano una markup XML, il gestore dati utilizza le informazioni specifiche sull'applicazione dell'attributo e genera XML come mostrato in Tabella 27..

Tabella 27. File di output XML per gli attributi che rappresentano una markup XML

Attributo che l'entità dell'oggetto di business XML rappresenta	**File di output XML**	**Esempio**	**Informazioni specifiche sull'applicazione**
Istruzione di elaborazione	`<?AttrValue?>`	`<?xml version="1.0"?>`	`type=pi`
DTD	`<!AttrValue>`	`<!DOCTYPE CUSTOMER "customer.dtd">`	`type=doctype`
Element	`<ElementName>... </ElementName>`	Per un documento XML basato su DTD: `<CUSTOMER>... </CUSTOMER>` Per un documento XML basato su documento dello schema: `<element name=CUSTOMER...> </element>`	`type=pcdata`
attributo XML	`AttrName= "AttrValue"`	Per un documento XML basato su DTD: `Seqno="1"` Per un documento XML basato su documento dello schema: `<element name=CUSTOMER... Seqno="1"...> </element>`	`type=attribute`
sezione CDATA	`<![CDATA[AttrValue]]>`	`<![CDATA [<HTML>Text</HTML>]]>`	`type=cdata`
Commento	`<!--CommentText -->`	`<!--Informazioni Cliente da applicazione sorgente A-->`	`type=comment`

Tabella 27. File di output XML per gli attributi che rappresentano una markup XML (Continua)

Attributo che l'entità dell'oggetto di business XML rappresenta	File di output XML	Esempio	Informazioni specifiche sull'applicazione
Posizione dello schema (con spazio dei nomi di destinazione)	<*elementName* xmlns="*URI_path*" xmlns:xsi= "http://www.w3.org/ 2001/XMLSchema- instance" xsi:schemaLocation= "*URI_for_schema schema_location*" ...	Consultare, Figura 28 a pagina 79	type= xsischemalocation
Posizione dello schema (nessuno spazio dei nomi di destinazione)	<*elementName* xmlns="*URI_path*" xmlns:xsi= "http://www.w3.org/ 2001/XMLSchema- instance" xsi:noNamespace SchemaLocation= "*schema_location*" ...	<order xmlns=″http://sampleDoc.org.ord″ xmlns:xsi=″http://www.w3.org/2001/ XMLSchema-instance″ xsi:noNamespaceSchemaLocation= ″order.xsd″> ... /<order>	type=xsinoNSlocation

- Per attributi che contengono dati carattere (PCDATA nei DTD) associati con un dato elemento, nessuna markup viene generata e solo il valore dell'attributo stesso è aggiunto al documento (assumendo che la tag notag esista nelle informazioni specifiche sull'applicazione).
- Per attributi che contengono dati carattere per un elemento o per un attributo, il gestore dati sostituisce qualsiasi carattere speciale con le appropriate sequenze escape (assumendo che la tag escape=true esista nelle informazioni specifiche sull'applicazione), come mostra Tabella 28.

  Tabella 28. Caratteri speciali e rappresentazioni XML

carattere speciale	sequenze escape XML
e commerciale (&)	&amp
minore di (<)	&lt
maggiore di (>)	&gt
virgoletta singola (')	&apos
virgolette doppie (")	&quot

Nota: Se un attributo rappresenta un elemento XML il cui valore contiene virgolette singole, virgolette doppie o i caratteri &, <, o >, allora l'attributo richiede l'elaborazione escape. Un attributo *non* verrà elaborato via escape finchè contiene il valore escape=true nelle sue informazioni specifiche sull'applicazione.Queste informazioni specifiche sull'applicazione devono essere collocate alla fine di qualsiasi altro testo.

Nota: L'ODA XML ignora gli attributi predefiniti e fissi sulle dichiarazioni di un elemento e di un attributo. Se una dichiarazione di un elemento o di un attributo specifica uno di questi attributi e l'istanza *non* contiene l'elemento o l'attributo, questa non è compilata dal gestore dati XML.

- Un attributo è saltato se si verifica una delle seguenti condizioni:
  - L'attributo ha il valore CxIgnore.

– Il nome dell'attributo è elencato nella tag `cw_mo_`, nelle informazioni specifiche sull'applicazione della definizione dell'oggetto di business.

Nessun elemento XML è generato per questi attributi.

- Attributi semplice `String` con un valore `CxBlank` sono inclusi nel documento XML come tag vuote. Per documenti XML basati sui DTD, sono utilizzate le doppie virgolette (`" "`) come l'equivalente `PCDATA` di `CxBlank`. Tuttavia, il gestore dati presume che nessun attributo complesso (uno il cui tipo sia un oggetto di business) abbia un valore di `CxBlank`.

4. Quando il gestore dati completa la conversione, restituisce il documento XML al richiedente.

Nota: Le informazioni del verbo in un oggetto di business vengono perse nella conversione in un documento XML. Per ulteriori informazioni su come conservare le informazioni del verbo, fare riferimento a "Verbi dell'oggetto di business" a pagina 38.

Conversione di documenti XML in oggetti di business

Questa sezione fornisce le seguenti informazioni su come il gestore dati XML converte i documenti XML in oggetti di business:

- "requisiti del documento XML"
- "Elaborazione dati serializzati"

requisiti del documento XML

Il gestore dati XML crea i seguenti presupposti riguardo a un documento XML:

- Il documento XML è ben strutturato.
- Il documento XML è conforme con i requisiti del parser SAX. Notare che il parser predefinito utilizzato dal gestore dati XML richiede che un documento XML includa una dichiarazione XML.
- La struttura di un documento XML deve corrispondere alla struttura del relativo oggetto di business corrispondente. Inoltre, l'ordine degli elementi nel documento deve corrispondere all'ordine degli attributi nell'oggetto di business.

Elaborazione dati serializzati

Durante la conversione di un documento XML in un oggetto di business, il gestore dati XML presume che l'oggetto di business segua la struttura del documento XML si adatti ai requisiti della definizione dell'oggetto di business descritti in "Requisiti per le definizioni oggetto di business" a pagina 33. Se non vi sono attributi nell'oggetto di business per un dato nome dell'elemento, il gestore dati XML restituisce un errore.

Per convertire un documento XML in un oggetto di business, il gestore dati XML effettua i seguenti passaggi:

1. Se il connettore richiedente viene passato in un oggetto di business al metodo di conversione, il gestore dati utilizza questo oggetto di business e continua. Se il richiedente *non* viene passato in un oggetto di business, il gestore dati determina il nome dell'oggetto di business e crea un oggetto di business per contenere i dati nel documento XML.

   Per determinare il nome dell'oggetto di business, il gestore dati richiama il gestore nome. Il gestore nome predefinito crea il nome dell'oggetto di business di livello superiore combinando l'attributo del meta oggetto `BOPrefix`, un segno di sottolineatura e il valore dell'elemento root. Ad esempio, se il documento

XML contiene <!DOCTYPE Customer> e BOPrefix l'attributo è MyApp, il nome risultante sarà MyApp_Customer. E' possibile fornire un gestore nome predefinito per configurare differenti comportamenti.

2. Il gestore dati recupera il valore dell'attributo del meta oggetto Parser per determinare quale parser SAX deve utilizzare per analizzare il documento XML. Per informazioni su quale parser SAX il gestore dati XML utilizzi, fare riferimento a "parser SAX" a pagina 31. Quando il gestore dati determina il nome del parser, crea un'istanza per il parser.
3. Il gestore dati registra il gestore evento (per un documento XML basato su DTD, registra anche il l'applicazione che risolve entità) con il parser. Il gestore evento è un metodo di richiamata che elabora ogni elemento e attributo XML.

   Nota: L'applicazione che risolve entità gestisce riferimenti esterni di entità nei documenti DTD. Se il gestore dati non trova un'opzione EntityResolver con un nome classe valido, utilizza com.crossworlds.DataHandlers.xml.DefaultEntityResolver. Quest'applicazione che risolve entità ignora tutte i riferimenti esterni.
4. Il gestore dati richiama il parser per analizzare il documento XML.
   - Il gestore dati determina se vi sono meta oggetti secondari (quelli i cui nomi sono elencati nella tag cw_mo_*label* delle informazioni specifiche sull'applicazione dell'oggetto di business). Il gestore dati *non* effettua l'elaborazione per compilare questi attributi dell'oggetto di business.
   - In base al tipo di elemento, il gestore evento del parser richiede le definizioni dell'oggetto di business per le proprietà dell'attributo ed elabora i dati dell'elemento data di conseguenza. L'esecuzione viene interrotta solo se si verificano errori gravi da parte del parser o quando un elemento nei dati XML non può essere trovato nella definizione dell'oggetto di business.
5. Quando l'oggetto di business è completo, il gestore dati lo restituisce al richiedente.

Nota: Per ogni elemento e attributo trovato in un documento XML, il gestore dati si aspetta di trovare un attributo nell'oggetto di business. Se non vi sono attributi nella definizione dell'oggetto di business per un dato elemento o nome dell'attributo, il gestore dati restituisce un errore. L'eccezione a questa regola è per gli attributi del tipo FISSATI, che non sono richiesti in una definizione dell'oggetto di business. Se gli attributi FISSI non sono presenti nella definizione dell'oggetto di business, l'esecuzione non si interrompe se viene trovato un attributo FISSO nel documento XML.

Configurazione del gestore dati XML

E' possibile personalizzare il gestore dati XML per:

- "Creazione di un gestore nome personalizzato XML"
- "Creazione di un'applicazione che risolve entità personalizzata" a pagina 91

Creazione di un gestore nome personalizzato XML

Il gestore dati XML richiede il gestore nome per estrarre il nome dell'oggetto business da un messaggio XML. Il gestore nome predefinito incluso con il gestore dati XML ricerca la tag:

```
<!DOCTYPE Name>
```

Da questa tag e dall'attributo del meta oggetto BOPrefix, il gestore dati genera il nome dell'oggetto di business. Il gestore dati XML determina quale gestore nome

per richiamare utilizzando il valore dell'attributo NameHandlerClass memorizzato nel meta oggetto del gestore dati. Se si desidera che il gestore nome funzioni in un modo differente, è necessario:

1. Creare un gestore nome predefinito estendendo la classe NameHandler.
2. Configurare il gestore dati XML per utilizzare la classe predefinita aggiornando il valore predefinito dell'attributo NameHandlerClass nel meta oggetto per il gestore dati XML.

Il seguente codice di esempio estende la classe DataHandler per creare un gestore dati personalizzato, CustomDataHandler, per il gestore dati XML:

```
package com.crossworlds.DataHandlers.xml;

// Dipendenze DataHandler
import com.crossworlds.DataHandlers.
     Exceptions.MalformedDataException;
import com.crossworlds.DataHandlers.NameHandler;
import com.crossworlds.DataHandlers.DataHandler;

// Classi Java
import java.io.*;
import java.lang.Exception;

/**********************************************************
 * CustomNameHandler class. Questa classe estende la classe Namehandler
 * e implementa il metodo:
 *      getBOName( Reader serializedData, String subType )
 * Il metodo getBOName contiene la logica per estrarre la classe pubblica BOName
**********************************************************/
CustomNameHandler estende NameHandler
{

         /**
          * Questo metodo genera il nome dell'oggetto di business dai
          * dati estratti dall'argomento 'serializedData'.
          * In tale caso, è compito del richiedente creare
          * la classe BOName.
          */

         public String  getBOName( Reader serializedData,
                                   String subType )
           throws MalformedDataException
         {
           // NameHandler utilizza il programma di traccia DataHandler. Se
           // DataHandler non è impostato,  NameHandler non viene eseguito.
           if (dh == null)
            return null;
           // Registrare un  messaggio
           dh.traceWrite(
           "Entering CustomNameHandler.getBOName for subtype '"
             + subType + "'.", 4);

           // Tale metodo analizza il documento XML ed estrae il
           // nome dell'oggetto di business dalla seguente tag nel
           // documento XML:
           //        <cml title=
           // Ad esempio, in:
           //        <cml title="cholestrol" id="cml_cholesterol">
           // il nome dell'oggetto di business è 'cholestrol'.

           // Registrare un messaggio
           dh.traceWrite(
             "La risoluzione del nome sarà effettuata utilizzando <cml title= ",4);
           String name = null;
```

```
            try
            {
              // Leggere la riga di dati dall'oggetto Reader
              LineNumberReader lineReader =
                new LineNumberReader( serializedData );
              serializedData.mark( 1000 );
              String line = lineReader.readLine();
              while ( line != null )
              {
                // ricercare <cml title= nella riga
                int start = line.indexOf("<cml title=");
                if ( start != -1 )
                {
                  start += 12;
                  // ricercare le virgolette finali per la tag affiancata
                  int end = line.indexOf('\"', start);

                  // estrarre il nome dalla riga
                  name = line.substring(start, end);
                  break;
                }
                line = lineReader.readLine();
              }

              if ( name == null || name.length() == 0 )
                throw new MalformedDataException(
                 "Error: can't determine the BusinessObject Name.");

            }

            catch(Exception e)
            {
              throw new MalformedDataException( e.getMessage() );
            }
            serializedData.reset();
            return name;
          }
}
```

Creazione di un'applicazione che risolve entità personalizzata

Il parser SAX utilizzato dal gestore dati XML richiede l'applicazione che risolve entità per trovare entità esterne (DTD riferiti) all'interno di un documento XML. L'applicazione che risolve entità inclusa nel gestore dati XML può ignorare riferimenti esterni o ricercarli su un file system locale. Se è necessario specificare un altro modo per trovare entità esterne, è necessario creare una classe che risolve entità personalizzata.

Il gestore dati XML determina quale applicazione che risolve entità deve essere richiamato utilizzando il valore dell'attributo `EntityResolver` memorizzato nel meta oggetto del gestore dati XML.

Capitolo 4. Gestore dati EDI

Il gestore dati IBM WebSphere Business Integration Server Express per EDI (Electronic Data Interchange), chiamato il gestore dati EDI, converte gli oggetti di business in documenti EDI e i documenti EDI in oggetti di business.

Questo capitolo descrive come il gestore dati EDI elabora i documenti EDI e come definire gli oggetti di business che devono essere elaborati dal gestore dati. E' possibile utilizzare questa guida per implementare gli oggetti di business che soddisfano i requisiti del gestore dati EDI. Inoltre, questo capitolo descrive come configurare il gestore dati XML. Questo capitolo contiene le sezioni seguenti:

Panoramica

Il gestore dati EDI è un modulo di conversione dati il cui ruolo principale è convertire gli oggetti business in documenti EDI e viceversa. Un documento EDI è un formato standardizzato per il trasporto di informazioni di business. Il gestore dati EDI supporta due standard di messaggio: X.12 e EDIFACT.

Il documento EDI è costituito da dati serializzati con `edi`. tipo MIME. Il meta-oggetto connettore di livello superiore predefinito (`MO_DataHandler_Default`) supporta il tipo MIME `edi`. Perciò, un connettore configurato per utilizzare il meta-oggetto del gestore dati `MO_DataHandler_Default` può richiamare il gestore dati EDI.

Attenzione

Il gestore dati EDI può unicamente elaborare un documento EDI contenente un singolo gruppo con un singolo tipo di transazione (il documento può contenere più transazioni dello stesso tipo). La maggior parte degli ambienti EDI gestiscono documenti con più gruppi e tipi di transazioni. L'adattatore IBM WebSphere Business Integration Server Express per TPI contiene una logica di suddivisione che consente all'adattatore di elaborare documenti EDI con più gruppi e tipi di transazioni. Nessun altro adattatore contiene tale logica quindi sebbene possano tecnicamente utilizzare il gestore dati EDI, solo l'adattatore per TPI è in grado di gestire situazioni in cui il documento EDI contiene più gruppi.

Il gestore dati analizza i dati del documento utilizzando separatori di documenti che identifica nel documento EDI. Se il gestore dati non riesce ad identificare i separatori nel documento, utilizza valori del separatore specificati dagli attributi nel meta-oggetto secondario associato al gestore dati EDI. Per ulteriori informazioni sul meta-oggetto secondario EDI, fare riferimento a "Configurazione del meta oggetto secondario del gestore dati EDI" a pagina 96.

Componenti del gestore dati EDI

Il gestore dati EDI utilizza un un gestore nome per estrarre il nome dell'oggetto di business da un messaggio EDI. Figura 29 illustra i componenti del gestore dati EDI e le relazioni intercorrenti.

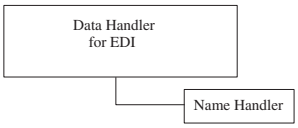

Figura 29. Componenti del gestore dati EDI

Il gestore dati invoca un'istanza del gestore nome basata sul valore dell'attributo `NameHandlerClass` nel meta oggetto secondario del gestore dati EDI:

- Se viene fornito un nome classe nell'attributo `NameHandlerClass`, il gestore dati EDI utilizza tale gestore nome per determinare il nome dell'oggetto di business.

  L'attributo `NameHandlerClass` nella versione del meta oggetto fornito con il prodotto fa riferimento al gestore nome EDI predefinito, che ottiene il nome dell'oggetto di business da creare dal file di ricerca del gestore nome che l'attributo `NameHandlerFile` specifica.
- Se non viene fornito alcun nome classe, il gestore dati registra un errore e genera un'eccezione.

Per ulteriori informazioni su come creare un gestore nome personalizzato, fare riferimento a "Personalizzazione del gestore dati EDI" a pagina 119.

Elaborazione dell'oggetto di business e del documento EDI

Il gestore dati EDI effettua le operazioni elencate in Tabella 29.

Tabella 29. Operazioni dati per il gestore dati EDI

operazione gestore dati	Per ulteriori informazioni
Riceve un oggetto di business dal richiedente, converte l'oggetto di business in un documento EDI e lo passa al richiedente.	"Conversione degli oggetti di business in documenti EDI" a pagina 107
Riceve un documento EDI dal richiedente, crea un oggetto di business e lo restituisce al richiedente.	"Conversione dei documenti EDI in oggetti di business" a pagina 110

Configurazione del gestore dati EDI

Per configurare il gestore dati EDI per l'utilizzo con un connettore, eseguire i seguenti passi:

- Creare un file di ricerca del gestore nome EDI per gli identificatori della transazione, numeri DUNS e nomi dell'oggetto di business.
- Immettere i valori appropriati per gli attributi del meta oggetto secondario EDI.

Ciascuno dei seguenti passi è descritto più dettagliatamente nelle sezioni seguenti.

Nota: Per utilizzare il gestore dati EDI, è necessario inoltre creare o modificare le definizioni dell'oggetto di business in modo che possano supportare il gestore dati. Per ulteriori informazioni, consultare "Definizioni degli oggetti di business per i documenti EDI" a pagina 98.

Creazione del file di ricerca del gestore nome

Il gestore dati EDI dipende da un file di ricerca del gestore nome EDI per determinare quale oggetto di business creare in base al messaggio EDI. Tale file di ricerca contiene le seguenti colonne:

- . Il identificatore di transazione (ID): questo valore identifica il tipo di documento EDI (ad esempio, 850). Non identifica solo i contenuti del documento EDI. Tuttavia, indica il tipo di informazioni che il documento contiene.
- (Facoltativo) Il numero di versione: questo è il Codice identificatore Version/Release/Industry che il gestore dati EDI utilizza per gestire più versioni dei documenti EDI
- . Il numero DUNS: il gestore dati EDI utilizza tale numero per identificare unicamente un partner commerciale.
- Il nome dell'oggetto di business associato: il gestore dati EDI utilizza il nome dell'oggetto di business per identificare l'oggetto di business EDI di livello superiore che dovrà creare.

Nota: Separare i campi del file di ricerca del gestore nome con caratteri tab.

Un esempio di file di ricerca del gestore nome con il numero versione opzionale è il seguente:

```
850             123465      X12_850A_Order
850             122227      X12_850B_Order

855             122227      X12_855A_Response
855             123465      X12_855A_Response
```

Un esempio di file di ricerca del gestore nome con il numero della versione è il seguente (i numeri della versione si trovano nella seconda colonna; in questo esempio il numero della versione è `004010`):

```
850             004010    111111      X12_850A_Order
855             004010    122227      X12_855A_Response
```

Nota: In questo esempio, "`X12_`" viene utilizzato come un prefisso comune per i nomi dell'oggetto di business di livello superiore. Questo prefisso *non* è obbligatorio. Si sceglie un prefisso di identificazione quando si creano i propri oggetti di business di livello superiore. Per ulteriori informazioni, consultare "Oggetto di business EDI di livello superiore" a pagina 99.

Per fornire al gestore dati EDI le informazioni sugli oggetti di business creati, è necessario:

- Assicurarsi che vi sia una voce nel file di ricerca del gestore nome per ogni combinazione del numero DUNS e dell'ID di transazione(e facoltativamente il numero della versione) per cui il gestore dati crea un oggetto di business. Assicurarsi che i valori della colonna siano separati.
- Impostare l'attributo del meta oggetto `NameHandlerFile` sul nome percorso completo per questo file di ricerca del gestore nome.

Nota: La proprietà Valore predefinito dell'attributo `NameHandlerFile` nella versione fornita del meta oggetto secondario contiene un valore. E' necessario assicurarsi, tuttavia, che il nome percorso nell'attributo `NameHandlerFile` specifichi il nome del proprio file di ricerca del gestore nome EDI. Quando si specifica il percorso su di un sistema Windows, è necessario evitare tutti i caratteri barra rovesciata (\) includendo una seconda barra rovesciata. Ad esempio:

```
c:\\home\\DataHandlers\\edi\\edi_xref
```

nomi percorso Linux non utilizzano barre rovesciate e quindi non vi è bisogno di evitarle:

```
/home/DataHandlers/edi/edi_xref
```

Il gestore dati EDI aggiorni le informazioni per tale file ogni volta che il file è aggiornato. Perciò, sceglie immediatamente valori nuovi o modificati, in modo da non dover riavviare alcun componente.

Configurazione del meta oggetto secondario del gestore dati EDI

Per configurare un gestore dati EDI, è necessario assicurarsi che le relative informazioni di configurazione siano fornite nel meta oggetto secondario del gestore dati EDI. Per il gestore dati EDI, IBM fornisce il meta oggetto secondario predefinito `MO_DataHandler_DefaultEDIConfig`. Ogni attributo in questo meta oggetto definisce una proprietà di configurazione per il gestore dati EDI. Tabella 30 descrive gli attributi in questo meta oggetto secondario.

Tabella 30. Attributi del meta oggetto secondario per il gestore dati EDI

Nome attributo	Descrizione	Valore predefinito fornito
`ClassName`	Nome della classe del gestore dati da caricare per l'utilizzo con il tipo MIME specificato. Il meta oggetto del gestore dati di livello superiore ha un attributo il cui nome corrisponde al tipo MIME specificato e il cui tipo è il meta oggetto secondario EDI (descritto dal Tabella 30).	`com.crossworlds. DataHandlers.edi. edi`
`DefaultVerb`	Nome del verbo da impostare nell'oggetto di business quando si effettua la conversione di un documento EDI in un oggetto di business. Se non esiste nessun valore per tale attributo, il gestore dati EDI non include un verbo nell'oggetto di business.	`Creare`
`DummyKey`	attributo Key; non utilizzato dal gestore dati ma richiesto dal sistema di integrazione business.	1
`ISA` (X.12 standard) `UNA` e `UNB` (EDIFACT standard)	Fornisce informazioni di posizione per i separatori in modo che il gestore dati EDI possa ottenere i valori dei separatori dal documento EDI stesso. E' necessario che il nome di tale attributo corrisponda al nome del primo segmento nel documento EDI, come segue: • Se i propri messaggi EDI seguono lo standard X.12, il documento EDI inizia con un segmento chiamato `ISA`; tale attributo di informazioni di posizione è chiamato `ISA`. • Se i propri messaggi EDI seguono lo standard EDIFACT, il documento EDI inizia con una notifica stringa del servizio `UNA` (facoltativo) ed un segmento iniziale chiamato `UNB`. Perciò, è necessario creare due attributi di informazioni di posizione in questo meta oggetto: `UNA` e `UNB`. Per ulteriori informazioni sui valori in tale attributo del meta oggetto, fare riferimento a 112.	*Nessuno*

Tabella 30. Attributi del meta oggetto secondario per il gestore dati EDI (Continua)

Nome attributo	Descrizione	Valore predefinito fornito
NameHandlerClass	Nome della classe da utilizzare per determinare il nome dell'oggetto di business dal contenuto di un documento EDI. Modificare la proprietà Valore predefinito di tale attributo se si intende creare un gestore dati personalizzato. Per ulteriori informazioni, consultare "Personalizzazione del gestore dati EDI" a pagina 119.	com.crossworlds. DataHandlers.edi. EdiNameHandler
NameHandlerFile	Nome completo del file di ricerca del gestore dati EDI, che contiene una tabella di ricerca del gestore nome per gli ID di transazione, un numero versione facoltativo, numeri DUNS e nomi dell'oggetto di business. Per ulteriori informazioni, consultare "Creazione del file di ricerca del gestore nome" a pagina 95.	Sistemi Windows: C:\\crossworlds\ edi\dbfile.txtsistemi Linux: /home/crossworlds/ edi/dbfile.txt
RELEASE_CHAR	Il carattere da utilizzare come un carattere escape nel valore dell'attributo value. Tale carattere escape è necessario se i separatori del documento EDI fanno parte dell'attuale valore dell'attributo. E' necessario far precedere il carattere nell'attuale valore da questo carattere escape. Ad esempio, se il valore di un attributo è "*dog?" e il separatore dell'elemento è l'asterisco, è necessario spostare l'asterisco nel valore dell'attributo, come segue: "?*dog??".	? (punto interrogativo)
SEPARATOR_ELEMENT	Il carattere o i caratteri utilizzati come separatore dell'elemento nel documento EDI.	* (asterisco)
SEPARATOR_COMPOSIT	Il carattere utilizzato come separatore misto nel documento EDI.	, (virgola)
SEPARATOR_REPEAT	Il carattere utilizzato come separatore ripeti nel documento EDI. E' utilizzato per dividere separatori misti ripetuti.	^ carattere di omissione (^)
SEPARATOR_SEGMENT	Il carattere utilizzato come separatore del segmento nel documento EDI. Se si desidera impostare il separatore del segmento in un carattere newline, è necessario spostare il carattere, come segue: • Sui sistemi Windows – documenti X12: \r\n – documenti EDIFACT: \n • Sui sistemi Linux: \n	~ (tilde)
ObjectEventId	Non viene utilizzato dal gestore dati ma richiesto dal sistema di integrazione business.	*Nessuno*

Il "Valore predefinito fornito" colonna in Tabella 30 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente nell'oggetto di business fornito. E' necessario verificare il proprio ambiente ed impostare le proprietà Valore predefinito di tutti gli attributi con i valori appropriati.

Nota: Utilizzare Business Object Designer Express per modificare le definizioni degli oggetti di business.

Per richiamare più configurazioni del gestore dati EDI, effettuare i seguenti passaggi:

- Copiare e rinominare il meta oggetto secondario predefinito EDI (che configura il gestore dati EDI per i documenti EDI nello standard X.12). Un approccio raccomandato per denominare un nuovo meta oggetto secondario è fornire

sottotipo del tipo MIME. Ad esempio, è possibile rinominare il meta oggetto secondario predefinito EDI per `MO_DataHandler_DefaultEDI_X12Config` e chiamare la relativa copia `MO_DataHandler_DefaultEDI_EDIFACTConfig`.

- Impostare i valori predefiniti degli attributi in ogni meta oggetto secondario EDI per configurare l'istanza del gestore dati.
- Creare attributi nel meta oggetto del gestore dati di livello superiore chiamato `edi_`*`sottotipo`*, dove *`sottotipo`* può essere uno degli standard EDI. Per gestire documenti EDI in uno standard X.12 o EDIFACT, è possibile creare due attributi nel meta oggetto di livello superiore: `edi_x12`, e `edi_edifact`. Ciascuno di questi attributi dovrebbe rappresentare i relativi meta oggetti secondari associati.

Per ulteriori informazioni su come configurare un gestore dati, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

Definizioni degli oggetti di business per i documenti EDI

Per utilizzare il gestore dati EDI, è necessario creare o modificare le definizioni dell'oggetto di business in modo che contengano i metadati che il gestore dati richiede e in modo che includano i campi corrispondenti a quelli nel messaggio EDI. Questa sezione fornisce le informazioni per creare le definizioni dell'oggetto di business necessarie per funzionare con il gestore dati EDI. In particolare, fornisce le seguenti informazioni:

- "Comprensione della struttura dell'oggetto di business EDI"
- "Creazione delle definizioni dell'oggetto di business per i documenti EDI" a pagina 106

Comprensione della struttura dell'oggetto di business EDI

Il gestore dati EDI utilizza le definizioni dell'oggetto di business quando converte oggetti di business o documenti EDI. Effettua la conversione utilizzando la struttura degli oggetti di business e le relative informazioni dell'applicazione. Figura 30 mostra la struttura degli oggetti di business che rappresenta un messaggio EDI.

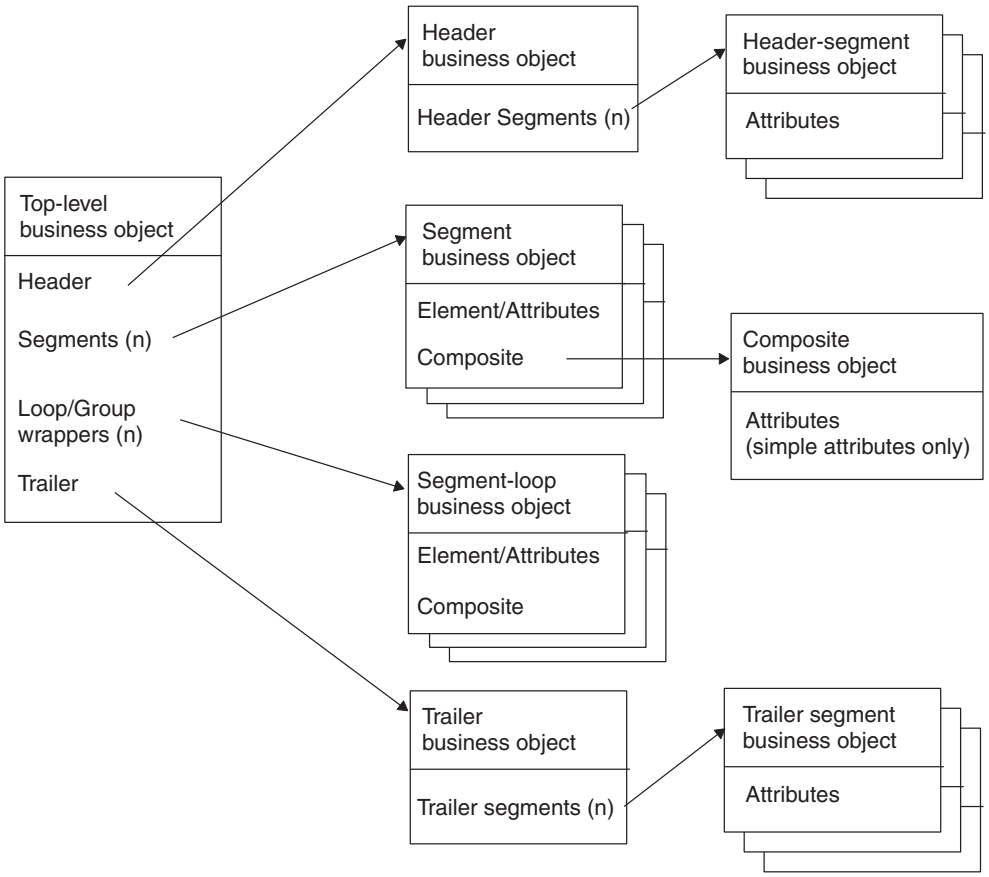

Figura 30. Struttura dell'oggetto di business per un messaggio EDI

Per assicurarsi che le definizioni dell'oggetto di business corrispondono ai requisiti del gestore dati EDI, utilizzare le indicazioni fornite per ciascuno dei seguenti oggetti di business:

- "Oggetto di business EDI di livello superiore"
- "Oggetto di business dell'intestazione" a pagina 102
- "Oggetto di business del segmento" a pagina 103
- "Oggetto di business misto" a pagina 105
- "Oggetto di business del loop del segmento" a pagina 105
- "Oggetto di business del trailer" a pagina 105

Oggetto di business EDI di livello superiore

Il gestore dati EDI prevede un oggetto di business di livello superiore per contenere le informazioni del messaggio EDI. Tabella 31 descrive come il gestore dati EDI interpreta le proprietà di un oggetto di business e descrive come impostare le proprietà quando si modifica un oggetto di business per l'utilizzo con il gestore dati EDI.

Tabella 31. Proprietà della definizione dell'oggetto di business di livello superiore EDI

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che tali definizioni degli oggetti di business inizino con un prefisso standard. Il nome dell'oggetto di business di livello superiore dipende dallo standard del messaggio, come mostrato di seguito: • Per un messaggio EDI che segue lo standard EDIFACT, il nome dell'oggetto di business ha la forma: *Prefisso BusObj + Message Type* • Per un messaggio EDI che segue lo standard X.12 standard, il nome dell'oggetto di business ha la forma: *Prefisso BusObj + Transaction Set Identifier Code*
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Non ci sono metadati che includono una tag `type`. E' possibile avere metadati con le tag `cw_mo` per indicare gli attributi che devono essere ignorati durante il processo di conversione.

Nota: Quando il gestore dati converte un documento EDI in un oggetto di business, identifica l'oggetto di business di livello superiore per il documento EDI mediante la tabella di ricerca del gestore nome. Per ulteriori informazioni, consultare "Creazione del file di ricerca del gestore nome" a pagina 95.

Come Figura 30 mostra, la definizione dell'oggetto di business di livello superiore contiene i seguenti attributi:

- Un attributo per rappresentare l'intestazione del documento EDI
- Tanti attributi quanti richiesti per rappresentare ogni segmento
- Tanti attributi quanti richiesti per rappresentare ogni loop o gruppi del segmento
- Un attributo per rappresentare il trailer del documento EDI

Attributo dell'intestazione: L'attributo dell'intestazione dell'oggetto di business di livello superiore EDI rappresenta un array a cardinalità singola che contiene le informazioni dell'intestazione. E' necessario che le informazioni dell'applicazione per tale attributo includano la seguente tag:

```
type=header
```

La proprietà Type per questo attributo contiene il nome dell'oggetto di business dell'intestazione. Per ulteriori informazioni sugli attributi dell'oggetto di business dell'intestazione, fare riferimento a "Oggetto di business dell'intestazione" a pagina 102.

Attributi del segmento: Ogni attributo del segmento dell'oggetto di business di livello superiore EDI rappresenta un array a cardinalità singola che contiene le informazioni del segmento. Tabella 32 mostra le proprietà dell'attributo per un attributo del segmento nella definizione dell'oggetto di business di livello superiore.

Tabella 32. Proprietà dell'attributo per un attributo del segmento nella definizione dell'oggetto di business di livello superiore EDI

Nome proprietà	Descrizione
Name	Il nome dell'attributo del segmento prende la forma: *Tag + Posizione* (se duplicato)
Type	Il tipo dell'attributo del segmento prende il nome della forma: *TopLevelBusObj + Tag* Questa proprietà dell'attributo contiene il nome dell'oggetto di business del segmento appropriato.
ContainedObjectVersion	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Relazione	Impostare su "contenimento".
Cardinalità	Se Max Use o Repetition sono impostati su "1" nelle specifiche della documentazione EDI, impostare il valore di queste proprietà dell'attributo su "1". Altrimenti, impostare tale proprietà su "N".
MaxLength	Sempre impostato su "1".
Key	Sempre impostato su "false".
Foreign key	Sempre impostato su "false".
Required	Se Status o Option sono impostati su "M" nelle specifiche della documentazione EDI, impostare tale proprietà dell'attributo su "true". Altrimenti, impostare tale proprietà su "false".
Valore predefinito	Non utilizzato dal gestore dati EDI.
Informazioni specifiche sull'applicazione	Impostare su: nome=*nome del segmento*

Nota: I campi Max Use, Repetition, Status, e Option fanno parte delle specifiche della documentazione EDI. Per ulteriori informazioni, fare riferimento alla documentazione EDI.

Per ulteriori informazioni sugli attributi dell'oggetto di business del segmento, fare riferimento a "Oggetto di business del segmento" a pagina 103.

Attributi del loop del segmento: Ogni attributo del loop del segmento rappresenta un array a cardinalità multipla contenente le informazioni del segmento. Tabella 33 mostra le proprietà dell'attributo per un attributo del loop del segmento nella definizione dell'oggetto di business di livello superiore.

Tabella 33. Proprietà dell'attributo per un attributo del loop del segmento nella definizione dell'oggetto di business di livello superiore EDI

Nome proprietà	Descrizione
Nome	Il nome dell'attributo del loop del segmento prende la forma: *Tag + Posizione* (se duplicato)
Type	Il tipo di un attributo del loop del segmento prende un nome della forma: *TopLevelBusObj + Loop/Group keyword + Tag* Tale proprietà dell'attributo contiene il nome dell'oggetto di business del loop del segmento appropriato. E' possibile che tale nome includa una parola chiave (come ad esempio Loop) per indicare lo scopo del segmento.

Tabella 33. Proprietà dell'attributo per un attributo del loop del segmento nella definizione dell'oggetto di business di livello superiore EDI (Continua)

Nome proprietà	Descrizione
ContainedObjectVersion	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Relazione	Impostare su "contenimento".
Cardinalità	Sempre impostato su "N".
MaxLength	Sempre impostato su "1".
Key	Sempre impostato su "false".
Foreign key	Sempre impostato su "false".
Required	Se Status o Option sono impostati su "M" nelle specifiche della documentazione EDI, impostare tale proprietà dell'attributo su "true". Altrimenti, impostare tale proprietà su "false". Impostare solo tale proprietà dell'attributo su "true" sul primo segmento in Loop/Group.
Valore predefinito	Non utilizzato dal gestore dati EDI.
Informazioni specifiche sull'applicazione	Includono: • name=*nome del primo segmento nel loop* • type=loop Ad esempio, le seguenti informazioni dell'applicazione identificano un loop del segmento il cui nome del primo segmento è AMT: AppSpecificInfo=name=AMT;type=loop

Nota: I campi Status e Option fanno parte delle specifiche della documentazione EDI. Per ulteriori informazioni, fare riferimento alla documentazione EDI.

Per ulteriori informazioni sugli attributi dell'oggetto di business del loop del segmento, fare riferimento a "Oggetto di business del loop del segmento" a pagina 105.

Attributo del trailer: L'attributo del trailer di un oggetto di business di livello superiore EDI rappresenta un array a cardinalità singola che contiene le informazioni del trailer. E' necessario che le informazioni dell'applicazione per tale attributo includano la seguente tag:

```
type=trailer
```

La proprietà Type per tale attributo contiene il nome dell'oggetto di business del trailer. Per ulteriori informazioni sugli attributi dell'oggetto di business del trailer, fare riferimento a "Oggetto di business del trailer" a pagina 105.

Oggetto di business dell'intestazione

Per contenere le informazioni dell'intestazione per il messaggio EDI, il gestore dati EDI prevede un oggetto di business dell'intestazione come primo attributo dell'oggetto di business di livello superiore. Tabella 34 descrive come il gestore dati EDI interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare queste proprietà quando si modifica l'oggetto di business per l'utilizzo con il gestore dati EDI.

Tabella 34. Proprietà per la definizione dell'oggetto di business dell'intestazione EDI

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che il nome dell'oggetto di business dell'intestazione includa il prefisso dell'oggetto di business. E' possibile inoltre che includa informazioni di identificazione come la parola chiave "intestazione".
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Non ci sono metadati che includono una tag type. E' possibile avere metadati con le tag cw_mo per indicare gli attributi che devono essere ignorati durante il processo di conversione.

Tale definizione dell'oggetto di business contiene un attributo per rappresentare ogni segmento di intestazione dell'intestazione. Le informazioni dell'applicazione per ogni attributo identificano il nome del segmento dell'intestazione. E' possibile che ogni segmento dell'intestazione contenga gli attributi semplici, a cardinalità singola o multipla.

Oggetto di business del segmento

Per contenere le informazioni del segmento per il messaggio EDI, il gestore dati EDI prevede un oggetto di business del segmento. Tabella 35 descrive come il gestore dati EDI interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare queste proprietà quando si modifica l'oggetto di business per l'utilizzo con il gestore dati EDI.

Tabella 35. Proprietà per la definizione dell'oggetto di business del segmento EDI

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che il nome della definizione dell'oggetto di business del segmento abbia la forma: *BusObj Prefix + Tag*
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Non ci sono metadati che includono una tag type. E' possibile avere metadati con le tag cw_mo per indicare gli attributi che devono essere ignorati durante il processo di conversione.

Come Figura 30 mostra, l'oggetto di business del segmento può contenere i seguenti attributi:

- Un attributo semplice (String) per rappresentare un elemento EDI
- Un attributo di array per rappresentare un elemento misto

Attributo semplice: E' necessario che ogni attributo semplice di un oggetto di business del segmento abbia le proprietà dell'attributo visualizzate in Tabella 36.

Tabella 36. Proprietà dell'attributo per gli attributi semplici

Nome proprietà	Descrizione
Name	E' necessario che ogni attributo dell'oggetto di business abbia una denominazione univoca.
Type	E' necessario che ogni attributo semplice dell'oggetto di business abbia un tipo String.
Cardinalità	Sempre impostato su "1".

Tabella 36. Proprietà dell'attributo per gli attributi semplici (Continua)

Nome proprietà	Descrizione
Key	Utilizzato solo per gli attributi semplici: è necessario che sia impostato per il *primo* attributo della stringa dell'oggetto di business.
MaxLength	Impostare la dimensione massima di tale attributo `String`. In un documento EDI, quando si inserisce un carattere separatore come parte dei dati attuali
Foreign key	Sempre impostato su `"false"`.
Required	Se `Status` o `Option` sono impostati su `"M"` nelle specifiche della documentazione EDI, impostare tale proprietà dell'attributo su `"true"`. Altrimenti, impostare tale proprietà su `"false"`.
Valore predefinito	Non utilizzato dal gestore dati EDI.

Nota: I campi `Repetition`, `Status` e `Option` fanno parte delle specifiche della documentazione EDI. Per ulteriori informazioni, fare riferimento alla documentazione EDI.

Attributo misto: Ogni oggetto di business misto è un array che contiene gli elementi di un elemento misto EDI. Tabella 37 mostra le proprietà dell'attributo per un attributo misto.

Tabella 37. Proprietà dell'attributo per un attributo misto in una definizione dell'oggetto di business del segmento EDI

Nome proprietà	Descrizione
Name	Il nome di un attributo misto prende la forma:*Tag* + *Position* (se duplicato)
Type	Il tipo dell'attributo del segmento prende il nome della forma: *`BusObj Prefix + Tag`* Tale proprietà dell'attributo contiene il nome dell'oggetto di business misto appropriato.
ContainedObjectVersion	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Relazione	Impostare su `"contenimento"`.
Cardinalità	Se `Repetition` è impostato su 1, impostare il valore di tale proprietà dell'attributo su `"1"`. Altrimenti, impostare tale proprietà su `"N"`.
MaxLength	Sempre impostato su `"1"`.
Key	Sempre impostato su `"false"`.
Foreign key	Sempre impostato su `"false"`.
Required	Se `Status` o `Option` sono impostati su `"M"`, impostare tale proprietà dell'attributo su `"true"`. Altrimenti, impostare tale proprietà su `"false"`.
Valore predefinito	Non utilizzato dal gestore dati EDI.
Informazioni specifiche sull'applicazione	*Nessuno*
Required Server Bound	Sempre impostato su `"false"`.

Nota: I campi `Repetition`, `Status` e `Option` fanno parte delle specifiche della documentazione EDI. Per ulteriori informazioni, fare riferimento alla documentazione EDI.

Per ulteriori informazioni, consultare "Oggetto di business misto" a pagina 105.

Oggetto di business misto

Per contenere le informazioni composite per un elemento nel messaggio EDI, il gestore dati EDI prevede un oggetto di business misto.

Nota: Gli elementi misti si trovano in genere nei documenti EDI che seguono lo standard EDIFACT. Tuttavia, è possibile che esistano anche nei documenti che seguono lo standard X.12.

Tabella 38 descrive come il gestore dati EDI interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare queste proprietà quando si modifica l'oggetto di business per l'utilizzo con il gestore dati EDI.

Tabella 38. Proprietà per la definizione dell'oggetto di business misto EDI

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che il nome della definizione dell'oggetto di business misto abbia la forma: *BusObj Prefix + Tag*
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Non ci sono metadati che includono una tag `type`. E' possibile avere metadati con le tag `cw_mo` per indicare gli attributi che devono essere ignorati durante il processo di conversione.

L'oggetto di business misto può contenere attributi semplici (`String`) o array.

Oggetto di business del loop del segmento

Per contenere informazioni per un loop o gruppo del segmento nel messaggio EDI, il gestore dati EDI prevede un oggetto di business del loop del segmento. Tabella 39 descrive come il gestore dati EDI interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare queste proprietà quando si modifica l'oggetto di business per l'utilizzo con il gestore dati EDI.

Tabella 39. Proprietà per la definizione dell'oggetto di business del loop del segmento EDI

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che il nome della definizione dell'oggetto di business del loop del segmento abbia la forma: *BusObj Prefix + Tag*
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Non ci sono metadati che includono una tag `type`. E' possibile avere metadati con le tag `cw_mo` per indicare gli attributi che devono essere ignorati durante il processo di conversione.

Oggetto di business del trailer

Per contenere le informazioni del trailer per il messaggio EDI, il gestore dati EDI prevede un oggetto di business trailer. Tabella 40 descrive come il gestore dati EDI interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare queste proprietà quando si modifica l'oggetto di business per l'utilizzo con il gestore dati EDI.

Tabella 40. Proprietà per la definizione dell'oggetto di business del trailer EDI

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che il nome dell'oggetto di business del trailer includa il prefisso dell'oggetto di business. E' possibile inoltre che includa informazioni di identificazione come la parola chiave "trailer".
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Non ci sono metadati che includono una tag type. E' possibile avere metadati con le tag cw_mo per indicare gli attributi che devono essere ignorati durante il processo di conversione.

Tale definizione dell'oggetto di business contiene un attributo per rappresentare ogni segmento di trailer del trailer. Le informazioni dell'applicazione per ogni attributo identificano il nome del segmento del trailer. E' possibile che i segmenti del trailer contengano attributi semplici, a cardinalità singola o a cardinalità multipla.

Creazione delle definizioni dell'oggetto di business per i documenti EDI

Esistono due modi per creare le definizioni dell'oggetto di business per un documento EDI:

- E' possibile utilizzare Edifec SpecBuilder per esportare la definizione di un documento EDI come definizione di un oggetto di business.
- e' possibile creare manualmente una definizione dell'oggetto di business per il documento.

Utilizzo di SpecBuilder per creare le definizioni dell'oggetto di business

SpecBuilder può funzionare come un programma di utilità di rilevamento oggetti, che crea definizioni dell'oggetto di business in base ad un documento EDI. SpecBuilder scrive le definizioni in un file della definizione dell'oggetto di business che può essere caricato nel sistema di integrazione business. SpecBuilder è un'applicazione di terzi prodotta e supportata da Edifecs Inc.. Per favore consultare la documentazione di SpecBuilder o il sito Web Edifecs per assistenza.

Nota: IBM non include lo strumento SpecBuilder come parte del suo prodotto. Lo strumento è, tuttavia, disponibile su di un CD Edifecs. Per ottenere una copia del CD Edifecs CD, contattare il proprio responsabile commerciale IBM o il supporto tecnico.

Creazione manuale delle definizioni dell'oggetto di business

Questa sezione descrive come creare manualmente le definizioni dell'oggetto di business per rappresentare documenti EDI. Utilizzare Business Object Designer Express per aggiungere o eliminare gli attributi dalla definizione dell'oggetto di business così come per modificare le proprietà dell'attributo, come richiesto.

Nota: E' possibile che la struttura di un documento EDI sia abbastanza complessa. Si raccomanda di utilizzare SpecBuilder per creare più definizioni dell'oggetto di business possibili.

Per definire un oggetto di business in base ad un documento EDI:

1. Creare la definizione dell'oggetto di business di livello superiore.

   Per ulteriori informazioni sulla struttura di tale oggetto di business di livello superiore, fare riferimento a "Oggetto di business EDI di livello superiore" a pagina 99.
2. Creare gli oggetti di business secondari per l'oggetto di business di livello superiore. Nell'oggetto di business di livello superiore, creare un attributo dell'oggetto secondario per gli oggetti di business mostrati in Tabella 41.

Tabella 41. Oggetti di business per il gestore dati EDI

Parte del documento EDI	Note	Oggetto business
Intestazione	E' possibile che tale intestazione contenga anche segmenti dell'intestazione.	"Oggetto di business dell'intestazione" a pagina 102
Segmenti	Un segmento può contenere anche elementi misti	"Oggetto di business del segmento" a pagina 103, "Oggetto di business misto" a pagina 105
Loop e Gruppi del segmento	Un loop del segmento consiste di segmenti ripetuti.	"Oggetto di business del loop del segmento" a pagina 105
Trailer	Questo trailer può contenere anche segmenti del trailer.	"Oggetto di business del trailer" a pagina 105

   Tenere presente quanto segue:
   - Il nome dell'attributo dell'oggetto di business non deve essere uguale al nome dell'elemento EDI. Le informazioni dell'applicazione sono utilizzate per specificare il nome dell'elemento.
   - Determinazione del tipo: `String` è un elemento contenuto cardinalità 1 senza contenuto dell'elemento o dichiarazione dell'elenco attributo associata. `BusinessObject` è un elemento contenuto a cardinalità `n`, o un elemento contenuto con il contenuto dell'elemento o con la specifica(o specifiche) dell'attributo associata.
   - Le informazioni dell'applicazione sono obbligatorie per molti degli attributi. Fare riferimento alle informazioni in "Oggetto di business EDI di livello superiore" a pagina 99 per ulteriori spiegazioni.
3. Creare un attributo dell'oggetto di business per ciascun elemento semplice. Per ulteriori informazioni, consultare "Attributo semplice" a pagina 103.
4. Creare gli oggetti di business secondari per ogni oggetto di business nidificato, come ad esempio segmenti dell'intestazione, segmenti del trailer e segmenti misti. Utilizzare le regole elencate in precedenza.

Conversione degli oggetti di business in documenti EDI

Per convertire un oggetto di business in un documento EDI, il gestore dati EDI effettua un loop tra gli attributi della definizione dell'oggetto di business di livello superiore. Elabora gli attributi in modo ricorrente, nell'ordine in cui appaiono nell'oggetto di business di livello superiore, scrivendo i valori dell'attributo come gli elementi del documento EDI.

Il gestore dati EDI elabora gli oggetti di business in un documento EDI nel modo seguente:

1. l gestore dati si inizializza impostando il misto, elemento, segmento, e ripeti separatori in base alle informazioni di configurazione nel meta oggetto secondario. Se non viene fornito nessun valore per una di queste opzioni di configurazione, il gestore utilizza le impostazioni predefinite codificate, come Tabella 42 e Tabella 45 mostrano.

Tabella 42. Valori predefiniti per i separatori dell'elemento e del segmento

Passaggio precedente		Separatore dell'elemento	Separatore del segmento
1	Ottenere il valore dell'attributo del meta oggetto corrispondente.	SEPARATOR_ELEMENT	SEPARATOR_SEGMENT
2	Se l'attributo del meta oggetto associato non è impostato, utilizzare un'impostazione predefinita codificata.	carattere più (+)	virgoletta singola (')

2. Il gestore dati esamina le informazioni delle applicazione nella definizione dell'oggetto di business di livello superiore per determinare se vi siano meta oggetti secondari (quelli i cui nomi sono elencati nella tag cw_mo_ delle informazioni dell'applicazione dell'oggetto di business). Il gestore dati *non* include questi attributi nel documento EDI. Per ulteriori informazioni sulla tag cw_mo_, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
3. Il gestore dati esegue un loop tra i rimanenti attributi nella definizione dell'oggetto di business di livello superiore. In base alla cardinalità di ogni attributo, il gestore dati determina che parte del documento EDI l'attributo rappresenta. Per ulteriori informazioni, consultare "Determinazione dei dati EDI associati con l'attributo" a pagina 109.
4. Una volta che il gestore dati identifica i dati EDI associati, è possibile effettuare i passaggi appropriati per scrivere i dati dell'attributo nel documento EDI:
   - Se un attributo rappresenta un segmento, il gestore dati controlla se questo è nullo. Se l'oggetto di business è nullo, il gestore dati ignora l'attributo. Se l'oggetto di business *non* è nullo, Il gestore dati effettua i passaggi per l'elaborazione del segmento. Per ulteriori informazioni, consultare "Elaborazione di un segmento" a pagina 110.
   - Per ogni oggetto di business secondario in un segmento che rappresenti un elemento misto, il gestore dati effettua un loop tra gli attributi (che dovrebbero essere tutti String) ed effettua i passaggi per l'elaborazione composita. Per ulteriori informazioni, fare riferimento a "Elaborazione di un elemento misto" a pagina 110.
5. Il gestore dati scrive il numero totale di segmenti che ha scritto per tale documento nel campo "numero dei segmenti" del documento. Il gestore dati determina la posizione di tale campo dalla tag seg_count nell'attributo ISA del meta oggetto secondario. Tale campo è spesso posizionato nel segmento SE. Per ulteriori informazioni sull'impostazione della tag seg_count, fare riferimento a "Ottenere informazioni di posizione" a pagina 112.
6. Quando il gestore dati completa la conversione, restituisce i dati serializzati al richiedente. Il gestore dati restituisce i dati come una stringa contenente il documento EDI.

Determinazione dei separatori del documento da inserire

Per convertire un oggetto di business in un documento EDI, è necessario che il gestore dati EDI inserisca correttamente i separatori nel documento EDI. Il gestore dati utilizza gli attributi nel meta oggetto secondario per determinare i valori da assegnare a questi separatori. Se non è fornito nessun valore per uno di questi attributi, il gestore dati utilizza le impostazioni predefinite codificate per il separatore.

Tabella 43 mostra i separatori EDI con i relativi attributi del meta oggetto corrispondenti ed i valori predefiniti codificati.

Tabella 43. Valori predefiniti per i separatori EDI

separatore EDI	Attributo del separatore in un meta oggetto secondario	Impostazione predefinita codificata
Separatore dell'elemento	SEPARATOR_ELEMENT	carattere più (+)
Separatore del segmento	SEPARATOR_SEGMENT	virgoletta singola (')
Separatore misto	SEPARATOR_COMPOSIT	due punti (:)
Separatore ripeti (solo documenti EDIFACT)	SEPARATOR_REPEAT	segno di omissione (^)

Attenzione:

Documenti EDI che seguono lo standard X.12*:* Affinché il gestore dati converta in maniera appropriata gli oggetti di business in un documento EDI, i valori degli attributi del separatore nel meta oggetto secondario *devono* corrispondere a quelli nel segmento ISA del documento EDI. Il gestore dati EDI *non* legge i dati nel segmento ISA per determinare i separatori del documento.

Documenti EDI che seguono lo standard EDIFACT: Affinché il gestore dati converta appropriatamente gli oggetti di business in un documento EDI, i valori degli attributi del separatore nel meta oggetto secondario deve essere impostato sui valori predefiniti, come determinato in Tabella 43. Gli attributi del separatore nel meta oggetto secondario predefinito (MO_DataHandler_DefaultEDIConfig) contiene valori validi per lo standard X.12. Assicurarsi di reimpostare questi valori dell'attributo predefinito sui valori definiti in Tabella 43.

Determinazione dei dati EDI associati con l'attributo

La struttura degli oggetti di business che contengono EDI viene determinata dalla specificazione del documento EDI. (Per informazioni su come creare tale struttura dell'oggetto di business, fare riferimento a "Creazione delle definizioni dell'oggetto di business per i documenti EDI" a pagina 106.) Il gestore dati EDI utilizza la cardinalità dell'attributo per determinare quale parte del documento EDI tale attributo rappresenta. In base alla cardinalità, il gestore dati effettua le seguenti azioni:

- Se l'attributo rappresenta un *array a cardinalità singola*, Il gestore dati controlla le informazioni dell'applicazione dell'attributo per determinare la parte del documento EDI associata a tale attributo.

  Il gestore dati controlla le informazioni dell'applicazione dell'attributo per una tag type (come type=header o type=trailer) che identifica il relativo scopo nella struttura EDI.

  – Se il gestore dati trova una tag type, elabora in modo ricorrente l'oggetto di business secondario.
  – Se il gestore dati *non* trova tale tag, il gestore dati assume che l'oggetto di business secondario rappresenta un segmento nel documento EDI ed effettua l'elaborazione come descritto in "Elaborazione di un segmento" a pagina 110.
- Se l'attributo rappresenta un *array a cardinalità multipla*, rappresenta un loop del segmento. Il gestore dati elabora in modo ricorrente ogni oggetto di business secondario come se fosse un array a cardinalità singola. Scrive un nuovo segmento nel documento EDI per ogni istanza dell'oggetto di business nell'array.
- Se l'attributo è del tipo String, il gestore dati genera un'eccezione poiché la struttura EDI richiede che tutti gli attributi dell'oggetto di business di livello superiore siano array a cardinalità singola o a cardinalità multipla.

Elaborazione di un segmento

Se un attributo rappresenta un segmento, le azioni che il gestore dati effettua dipendono se l'attributo è nullo:

- Se il valore dell'attributo è nullo, il gestore dati ignora l'attributo; *non* lo include nel documento EDI.
- Se il valore dell'attributo non è nullo, il gestore dati effettua i seguenti passi di elaborazione:
  - Analizza le informazioni dell'applicazione dell'oggetto di business per il nome del segmento, una tag della forma:

    `name=`*`nome_segmento`*
  - Aggiunge il separatore del segmento al documento EDI.
  - Aggiunge il nome del segmento al documento EDI, inserisce i caratteri escape richiesti.
  - Per ogni oggetto di business secondario (sia a cardinalità singola che multipla), il gestore dati aggiunge il separatore dell'elemento al documento EDI ed elabora l'oggetto di business secondario come misto (fare riferimento a "Elaborazione di un elemento misto"). Per un attributo a cardinalità multipla, il gestore dati elabora ciascun oggetto di business secondario nell'ordine in cui si trova.
  - Per ogni attributo `String` che *non* è nullo, il gestore dati aggiunge il separatore dell'elemento ed il valore dell'attributo, inserendo caratteri escape se richiesto.

Elaborazione di un elemento misto

Per ogni oggetto di business secondario che rappresenta un elemento misto, il gestore dati effettua un loop tra gli attributi (che dovrebbero essere tutti `String`) ed effettua i seguenti passi di elaborazione:

- Analizza i dati dell'attributo, aggiungendo ogni carattere escape necessario.
- Aggiunge il valore dell'attributo con ogni carattere escape al documento EDI.
- Aggiunge il separatore misto al documento.

Conversione dei documenti EDI in oggetti di business

Per convertire un documento EDI in un oggetto di business, il gestore dati EDI effettua un loop tra gli attributi della definizione dell'oggetto di business di livello superiore. Ottiene il nome dell'oggetto di business da creare, quindi elabora gli attributi in modo ricorrente, nell'ordine in cui questi appaiono nell'oggetto di business di livello superiore e nei relativi secondari, assegnando valori dell'elemento dal documento EDI all'oggetto di business.

Il gestore dati EDI elabora un documento EDI in un oggetto di business nel modo seguente:

1. Il gestore dati imposta ogni proprietà passata attraverso l'oggetto di configurazione facoltativo. Tali informazioni dovrebbero essere passate attraverso l'argomento `config` del metodo `getBO()`.
2. Il gestore dati viene inizializzato per preparare la lettura del documento EDI. Per ulteriori informazioni, consultare "Inizializzazione del gestore dati" a pagina 111.
3. Se il gestore dati non riceve un oggetto di business dal richiedente, è necessario crearne uno in base al nome dell'oggetto di business che si trova nel file di

ricerca del gestore nome. Per ulteriori informazioni, consultare “Determinazione del nome dell’oggetto di business” a pagina 116.

4. Appena il gestore dati ha accesso ad un’istanza dell’oggetto di business di livello superiore, il gestore dati immette in tale oggetto di business e nei relativi oggetti secondari i dati del documento EDI. Per ulteriori informazioni, consultare “Compilare l’oggetto di business” a pagina 117.
5. Una volta che il gestore dati ha completato la conversione, restituisce l’oggetto di business di livello superiore al richiedente. Il gestore dati restituisce l’intera gerarchia, l’oggetto di business di livello superiore e tutti i relativi oggetti secondari.

Inizializzazione del gestore dati

Perché venga inizializzato per la conversione di un documento EDI in un oggetto di business, il gestore dati EDI effettua i seguenti passaggi:

1. Controlla che l’oggetto `Reader` contenente i dati serializzati supporti l’operazione `mark()`.
2. Inizia l’analisi del documento EDI per ottenere il primo nome del segmento, i separatori, l’ID di transazione ed il numero DUNS.

Ciascuno dei seguenti passi è descritto più dettagliatamente nelle sottosezioni seguenti.

Verifica dell’oggetto Reader

. Il E’ necessario che il gestore dati sia in grado di contrassegnare una particolare posizione nel documento EDI e quindi di ritornare in seguito a tale posizione. Poiché il documento EDI viene passato al gestore dati EDI come un oggetto `Reader`, è necessario che tale oggetto `Reader` sia in grado di supportare l’operazione `mark()`.

Come primo passo di inizializzazione, perciò, il gestore dati EDI verifica che l’oggetto `Reader` ricevuto supporti l’operazione `mark()`. Se non la supporta, il gestore dati registra un errore e genera un’eccezione. Si raccomanda che tutti i dati serializzati siano passati nel gestore dati EDI in un oggetto `StringReader`.

Nota: Per ulteriori informazioni sull’oggetto `Reader` e sull’operazione `mark()`, fare riferimento alla sezione Note nella descrizione di “getBO() - public” a pagina 206.

Determinazione dei separatori del documento da leggere

Per convertire un documento EDI in un oggetto di business, è necessario che il gestore dati EDI legga correttamente i separatori nel documento EDI. Il gestore dati analizza il documento per ottenere tali separatori. Poiché i primi tre caratteri del documento EDI sono noti, il gestore dati analizza per primi questi tre. Il gestore dati legge i primi tre caratteri per determinare se rappresentano:

- La notifica stringa del servizio `UNA` (solo documenti EDIFACT)

  La notifica stringa del servizio UNA contiene i separatori del documento da utilizzare.
- Il nome del segmento iniziale

  E’ necessario che il gestore dati analizzi il documento per ottenere i separatori del documento ed altre informazioni di posizioni.

Verifica della notifica stringa del servizio UNA: La notifica stringa del servizio `UNA` è il primo elemento facoltativo nei documenti EDI che seguono lo standard EDIFACT. Tale stringa del servizio è costituita da sei caratteri alfanumerici nel

seguente ordine:

Separatore del componente
Separatore dell'elemento
segno decimale
carattere Release
Separatore ripeti (solo versione 4 della sintassi)
Separatore del segmento

Se i primi tre caratteri del documento EDI sono "UNA", il gestore dati utilizza i valori che la stringa del servizio UNA specifica per interpretare il documento EDI. Questi valori del separatore hanno la precedenza su ogni altra impostazione del separatore nel documento EDI, incluso qualsiasi attributo di informazioni di posizione UNA o UNB del meta oggetto secondario.

Nota: Per un documento EDI con una notifica stringa del servizio UNA, il gestore dati ottiene l'ID di transazione ed il numero DUNS dall'attributo di informazioni di posizione UNA del meta oggetto secondario. Per ulteriori informazioni, fare riferimento alla prossima sezione.

Ottenere informazioni di posizione: Se i primi tre caratteri del documento EDI *non* sono "UNA", il gestore dati assume che rappresentano il nome del segmento iniziale. Il gestore dati assume che i segmenti iniziali facciano parte dell'intestazione e abbiano nomi che sono lunghi esattamente tre caratteri. Se non esiste alcuna notifica stringa del servizio UNA, è necessario che il gestore dati ottenga i separatori del documento dal documento EDI stesso. Il gestore dati prosegue l'analisi del documento EDI, effettuando i seguenti passaggi:

- Legge il quarto carattere del documento EDI per determinare il separatore dell'elemento.

  Se il gestore dati non è in grado di determinare il separatore dell'elemento, utilizza il valore dell'attributo SEPARATOR_ELEMENT del meta oggetto secondario. Il valore fornito per SEPARATOR_ELEMENT è un asterisco (*). Se, per qualsiasi ragione, il gestore dati non è in grado di ottenere il separatore dell'elemento dal meta oggetto secondario, utilizza il relativo codice predefinito del segno più (+).
- Ottiene le informazioni di posizione dall'attributo nel meta oggetto secondario

  Le informazioni di posizione includono i separatori (del segmento, misti e separatori ripeti), ID di transazione e numeri DUNS. Solitamente il gestore dati può determinare tali informazioni di posizione analizzando il documento EDI.

  Per favorire il gestore dati a trovare le informazioni di posizione nel documento EDI, il meta oggetto secondario associato con il gestore dati EDI (MO_DataHandler_DefaultEDIConfig per impostazione predefinita) contiene un attributo contenente tali informazioni di posizione. Il nome di tale attributo delle informazioni di posizione corrisponde al nome del primo elemento nel documento EDI, come segue:
  - Per lo standard X.12, il documento EDI inizia con un segmento chiamato "ISA". Perciò, il gestore dati cerca un attributo ISA nel meta oggetto secondario.
  - Per lo standard EDIFACT, la maggior parte dei documenti EDI inizia con un segmento chiamato "UNB". Tuttavia, prima dovrebbe trovarsi una notifica stringa del servizio facoltativa UNA. Perciò, il gestore dati cerca un attributo nel meta oggetto secondario che corrisponda all'elemento iniziale (UNA, UNB o qualunque sia il nome del primo elemento). Il gestore dati non ha modo di

sapere se una notifica stringa del servizio UNA esiste finchè non inizia l'analisi del documento EDI. Perciò, è necessario che entrambi gli attributi UNA e UNB esistano nel meta oggetto secondario.

Nota: Per impostazione predefinita, il meta oggetto MO_DataHandler_DefaultEDIConfig configura il gestore dati EDI per lo standard X.12. Perciò, fornisce un attributo ISA. Se i propri messaggi seguono lo standard EDIFACT, è necessario aggiungere o gli attributi UNA e UNB al meta oggetto secondario predefinito o creare un meta oggetto secondario distinto per lo standard EDIFACT, che contenga gli attributi UNA e UNB invece di un attributo ISA.

L'attributo delle informazioni di posizione specifica le informazioni con le serie di tag che Tabella 44 mostra.

Tabella 44. Informazioni del documento EDI nell'attributo delle informazioni di posizione

Informazioni del documento EDI	Tag dell'attributo	Descrizione	Required?
Separatore del segmento	lunghezza	Specifica la lunghezza del primo segmento (in numeri di caratteri), includendo il nome del segmento ma escludendo il separatore del segmento. **Nota:** Se il separatore del segmento è seguito da un carattere newline (\n), tale carattere newline viene incluso come parte del separatore del segmento. Ad esempio, se il separatore è dato come '\n (virgoletta singola e carattere newline insieme), allora entrambi i caratteri sono specificati come il separatore.	Sì
Calcolo del segmento	seg_count	Specifica la posizione relativa del campo contenente il numero di segmenti scritti nel documento EDI (durante la conversione oggetto di business in EDI). Per ulteriori informazioni sull'utilizzo di questa tag, fare riferimento a "Conversione degli oggetti di business in documenti EDI" a pagina 107.	Sì
Separatore misto	cs	Fornisce la posizione relativa del separatore misto.	No
Separatore ripeti	rs	Fornisce la posizione relativa del separatore ripeti.	No
Identificatore della transazione	tid	Fornisce la posizione relativa dell'ID di transazione.	Sì
numero DUNS	duns	Fornisce la posizione relativa del numero DUNS.	Sì
Numero versione/rilascio	versione	Fornisce la posizione relativa del numero di versione del gruppo/messaggio funzionale.	No

Come Tabella 44 indica, è necessario che l'attributo delle informazioni di posizione fornisca i valori per le tag seg_count, length, tid e duns e facoltativamente per la tag version. Non è obbligatorio specificare valori per le tag cs e rs. Tuttavia, se entrambe queste tag vengono omesse e il gestore dati analizza i documenti EDI contenenti gli elementi misti, il gestore dati utilizza il passaggio precedente mostrato in Tabella 45 per ottenere un valore.

Tabella 45. Valori predefiniti per i separatori misto e ripeti

Passaggio precedente		Separatore misto	Separatore ripeti
1	Ottenere il valore dell'attributo del meta oggetto corrispondente.	SEPARATOR_COMPOSIT	SEPARATOR_REPEAT
2	Se l'attributo del meta oggetto associato non è impostato, utilizzare un'impostazione predefinita codificata.	due punti (:)	segno di omissione (^)

Le tag cs, rs, tid e duns utilizzano il seguente formato per indicare la posizione relativa in un documento EDI:

tagname=seg_name+elem_pos+compos_pos

dove:

- seg_name è il nome del segmento in cui l'informazione è ubicata.
- elem_pos è la posizione dell'elemento nel segmento *seg_name*. La numerazione delle posizioni dell'elemento inizia con "1" (non zero).
- compos_pos è facoltativo; specifica la posizione dell'elemento nell'elemento misto *elem_pos* (assumendo che *elem_pos* si riferisca ad un elemento misto). La numerazione per le posizione composite inizia con "1" (non zero).

La tag seg_count utilizza il seguente formato per indicare la propria posizione relativa in un documento EDI:

seg_count=*seg_name+elem_pos*

dove seg_name e *elem_pos* sono come prima descritti; ossia, la specifica seg_count non include *mai* un valore *compos_pos*.

Nota: Non includere spazi tra i valori seg_name, *elem_pos* e *compos_pos* ed il segno più.

Figura 31 mostra un esempio del documento EDI che utilizza lo standard X.12. Per migliorare la leggibilità di questo documento EDI, l'esempio inserisce caratteri newline alla fine di ogni segmento.

```
ISA*00*0000000000*02*XXXX*cw*ldtp3*cw*ld*970106*1525*U*00200*0000000100*0*P*<
GS*AA*ldtp3*ld*20010424*1525*142*X*004010
ST*846*001420001
SE*2*001420001
GE*1*142
IEA*1*0000000100
```

Figura 31. Esempio di documento EDI nello standard X.12

Per ottenere le informazioni di posizione da un documento EDI che segue lo standard X.12, il gestore dati EDI esegue i seguenti passaggi:

1. Localizza un attributo nel meta oggetto secondario del gestore dati che corrisponde al nome del primo segmento.

   Per l'esempio del documento EDI in Figura 31, il meta oggetto secondario ha bisogno di contenere un attributo ISA (poiché il nome del primo segmento è ISA).
2. Da questo attributo del meta oggetto, ottiene le informazioni di posizione.

   Nell'esempio corrente, l'attributo ISA contiene le seguenti informazioni di posizione:

   ```
   length=77;tid=ST+1;duns=ISA+6;seg_count=SE+1
   ```

   o (Se la versione è inclusa in dbfile.txt) allora

   ```
   length=77;tid=ST+1;version=GS+8;duns=ISA+6;seg_count=SE+1
   ```
3. Continua l'analisi del prima segmento del documento per determinare il separatore del segmento. Il gestore dati assume che il separatore del segmento è alla fine del primo segmento quindi effettua i seguenti passaggi:

- Si sposta alla fine di questo segmento, in base alla lunghezza del primo segmento, fornito dalla tag di lunghezza nell'attributo delle informazioni di posizione del meta oggetto secondario.
- Ottiene il carattere da questa posizione come il separatore del segmento.

Nota: A causa dell'algoritmo che il gestore dati utilizza per localizzare il separatore del segmento, tale separatore non può essere impostato su un carattere alfanumerico. Nell'esempio del passaggio precedente 2, la tag di lunghezza specifica una lunghezza del segmento di 77 carattere, che indica che il separatore del segmento nel documento Figura 31 è il carattere newline (ritorno a capo). Perciò, il gestore dati interpreta ogni carattere newline come un separatore del segmento.

4. Continua l'analisi del primo segmento per determinare L'ID transazione, in base alla tag `tid=` nell'attributo delle informazioni di posizione del meta oggetto secondario.

   Il documento Figura 31 segue lo standard X.12. Tale esempio *non* include alcun separatore misto. Perciò, il relativo attributo `ISA` (in 2) non include separatori misti (`cs`) o separatori ripeti (`rs`). Tale attributo include la tag `tid`, che specifica che l'ID di transazione si trova come primo elemento nel segmento chiamato `ST`; perciò, l'ID di transazione per il documento Figura 31 è `846`.
5. Continua l'analisi del primo segmento per trovare `version` (se la tag facoltativa `version` è specificata in `dbfile.txt`).

   In Figura 31 `version` è l'ottavo elemento del segmento `GS`: `004010`.
6. Analizza il documento per trovare il numero DUNS. Se il gestore dati non può trovare il numero DUNS, viene registrato un errore e generata un eccezione.

   In 2, la tag `duns` specifica che il numero DUNS si trova come sesto elemento nel segmento chiamato `ISA`; perciò, il numero DUNS per il documento Figura 31 è `ldtp3`.

Per ottenere le informazioni di posizione da un documento EDI che segue lo standard EDIFACT, il gestore dati effettua gli stessi passaggi descritti per l'analisi di un documento EDI che segue lo standard X.12. Le uniche differenze principali sono:

- E' necessario che il gestore dati localizzi un attributo `UNB` nel meta oggetto secondario del gestore dati (poiché il nome del primo segmento è `UNB`, non `ISA`).
- E' più probabile che il gestore dati cerchi separatori misti nell'analisi di un documento EDIFACT EDI (i separatori misti si trovano raramente nei documenti X.12). Quando viene trovato un separatore misto, è necessario che il gestore dati analizzi il documento per i separatori misti ed i separatori ripeti:
  - Se il documento EDI contiene separatori misti, è necessario che l'attributo delle informazioni di posizione del meta oggetto secondario contenga almeno la tag `cs`. E' possibile, inoltre, che contenga la tag `rs`, se il documento utilizza un separatore ripeti non predefinito. Il gestore dati determina il separatore misto ed il separatore ripeti in base alle informazioni di posizione fornite dalle tag `cs` e `rs`. Se il gestore dati non può determinare i separatori, utilizza uno dei valori predefiniti elencati in Tabella 45.
  - Se il documento EDI *non* contiene separatori misti, l'attributo delle informazioni di posizione non ha bisogno di contenere le tag `cs` o `rs`.

La seguente riga è un frammento di un documento EDI che segue lo standard EDIFACT:

```
ST*st_child_value_1*,*st_grand_child_val_11,st_grand_child_val_12^
st_grand_child_val_13,st_grand_child_val_14*st_child_value_4*
st_grand_child_val_21,st_grand_child_val_22
```

Figura 32. Frammento del documento EDI di esempio con un separatore misto

Se il relativo primo segmento è chiamato "UNB", allora il meta oggetto secondario contiene un attributo UNB comprendente la seguente tag cs:

```
cs=ST+2;
```

Questa tag cs specifica che il separatore misto si trova come secondo elemento nel segmento chiamato ST; perciò, il gestore dati interpreta la virgola (,) come il separatore misto. Supporre che il documento EDI in cui tale frammento si trova *non* abbia specificato un separatore ripeti; allora il gestore dati utilizza il valore predefinito del carattere di omissione (^). Perciò, l'attributo UNB nel meta oggetto secondario che il documento utilizza *non* ha bisogno di contenere una tag rs per specificare il separatore ripeti. Senza tag rs, il gestore dati assume che il separatore ripeti contenga il suo valore predefinito. Quando il gestore dati incontra un carattere di omissione (^), lo interpreta come il separatore ripeti.

Per definire un separatore ripeti non predefinito, è necessario che il documento EDI includa il carattere non predefinito in un campo (generalmente nell'intestazione) ed è necessario che l'attributo delle informazioni di posizione nel meta oggetto secondario associato includa la tag rs per indicare la posizione di tale campo.

Determinazione del nome dell'oggetto di business

Un gestore dati può ricevere i dati serializzati in uno di questi due modi:

- Riceve i dati serializzati *e* un oggetto di business vuoto: il gestore dati immette nell'oggetto di business i dati serializzati.
- Riceve *solo* i dati serializzati: il gestore dati deve creare l'oggetto di business prima di poterlo compilare con i dati serializzati.

Nota: Se il gestore dati riceve un oggetto di business, passa direttamente al passaggio descritto in "Compilare l'oggetto di business" a pagina 117.

Se il gestore dati *non* riceve un oggetto di business, è necessario che il gestore dati determini il tipo di oggetto di business da creare. Il gestore dati richiama il gestore nome, che effettua i seguenti passaggi:

1. Apre il file di ricerca del gestore nome EDI in base al nome dato nell'attributo NameHandlerFile del meta oggetto secondario. E' necessario che questo file di ricerca del gestore nome esista già. Se l'apertura del file non avviene, il gestore nome genera un'eccezione. Per ulteriori informazioni, consultare "Creazione del file di ricerca del gestore nome" a pagina 95.
2. Controlla se il file di ricerca del gestore nome sia stato modificato dall'ultima volta che è stato letto. Se è stato modificato, legge nuovamente i contenuti nella tabella di ricerca del gestore nome memorizzata.
3. In base all'ID transazione e al numero DUNS (che sono determinati nella fase di inizializzazione), ricerca il nome dell'oggetto di business di livello superiore EDI associato a questo documento EDI nella tabella di ricerca del gestore nome.

Se la ricerca del nome dell'oggetto di business ha esito negativo, il gestore dati registra un errore e genera un'eccezione. Se invece la ricerca ha esito positivo, il gestore dati crea un oggetto di business del tipo specificato per contenere i dati.

Nota: Questi passaggi descrivono il comportamento del gestore nome predefinito fornito con il gestore dati EDI. Per ulteriori informazioni su come creare un gestore nome personalizzato, fare riferimento a "Personalizzazione del gestore dati EDI" a pagina 119.

Compilare l'oggetto di business

Non appena i separatori EDI sono stati determinati e l'oggetto di business di livello superiore è stato creato, il gestore dati effettua i seguenti passaggi per compilare l'oggetto con i dati serializzati:

1. Se l'attributo del meta oggetto `DefaultVerb` è impostato, il gestore dati imposta il verbo nell'oggetto di business sul valore che `DefaultVerb` specifica. Il valore fornito per `DefaultVerb` è `Create`. Altrimenti, il gestore dati assume che nessun verbo necessita di essere impostato.
2. Il gestore dati determina se ci sono meta oggetti secondari (quelli i cui nomi sono elencati nella tag `cw_mo_` delle informazioni specifiche dell'applicazione dell'oggetto di business). Il gestore dati *non* effettua l'elaborazione per popolare questi attributi dell'oggetto di business. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
3. Il gestore dati esegue un loop tra i rimanenti attributi nella definizione dell'oggetto di business di livello superiore. In base alla cardinalità di ogni attributo, il gestore dati determina che parte del documento EDI l'attributo rappresenta. Per ulteriori informazioni, consultare "Determinazione dell'attributo associato ai dati EDI".
4. Non appena il gestore dati identifica l'attributo associato con i dati EDI correnti, il gestore dati effettua i passaggi appropriati per scrivere i dati EDI per tale attributo. Il gestore dati analizza i dati EDI in base ai separatori (che sono determinati nella fase di inizializzazione). Per ulteriori informazioni, consultare "Analisi del documento EDI" a pagina 118.

Non appena il gestore dati ha compilato tutti gli attributi dell'oggetto di business di livello superiore, può effettuare un controllo facoltativo per assicurarsi che tutti i dati EDI siano stati analizzati.

Determinazione dell'attributo associato ai dati EDI

La struttura degli oggetti di business che contengono EDI viene determinata dalla specificazione del documento EDI. (Per informazioni su come creare tale struttura dell'oggetto di business, fare riferimento a "Creazione delle definizioni dell'oggetto di business per i documenti EDI" a pagina 106.) Il gestore dati EDI utilizza la cardinalità dell'attributo per determinare se ogni attributo rappresenta la parte EDI corrente del documento EDI. In base alla cardinalità, il gestore dati effettua le seguenti azioni:

- Se l'attributo rappresenta un *array a cardinalità singola*, il gestore dati controlla le informazioni dell'applicazione dell'attributo per determinare la parte del documento EDI associata a tale attributo, come segue:
  - Esamina i metadati nelle informazioni specifiche dell'applicazione dell'attributo per determinare il tipo di oggetto di business da creare.
  - Crea l'oggetto di business indicato dalle informazioni dell'applicazione dell'attributo, come Tabella 46 mostra.

Tabella 46. Informazioni dell'applicazione e oggetti di business EDI associati

Informazioni specifiche sull'applicazione	Oggetto di business secondario
`type=intestazione`	Oggetto di business dell'intestazione

Tabella 46. Informazioni dell'applicazione e oggetti di business EDI associati (Continua)

Informazioni specifiche sull'applicazione	Oggetto di business secondario
name=nome segmento	Oggetto di business del segmento
name=nome del primo segmento nel loop;type=loop	oggetto di business del loop del segmento
type=trailer	Oggetto di business del trailer

  - Elabora in modo ricorrente questo nuovo oggetto di business secondario effettuando un loop tra i relativi attributi nell'ordine in cui si trovano nella definizione dell'oggetto di business.
- Se l'attributo rappresenta un *array a cardinalità multipla*, rappresenta un loop del segmento. Il gestore dati elabora in modo ricorrente ogni oggetto di business secondario come se fosse un array a cardinalità singola. Crea un nuovo oggetto di business nell'array per ogni istanza del loop che si verifica nel documento EDI.
- Se l'attributo è del tipo `String`, il gestore dati genera un'eccezione poiché la struttura EDI richiede che tutti gli attributi dell'oggetto di business di livello superiore siano array a cardinalità singola o a cardinalità multipla.
- Se il gestore dati non può determinare quale attributo associare al segmento, visualizza un messaggio di errore riguardo ad un nome del segmento sconosciuto. Se si aggiunge la proprietà di configurazione `doStrictCheck` al meta oggetto secondario del gestore dati EDI e si imposta il relativo valore su `true`, il gestore dati lancerà la seguente eccezione:

  ```
  Nome segmento <nome del segmento> non ha corrispondenze
  Attributo dell'oggetto di business!
  ```

  Per ulteriori informazioni sul meta oggetto secondario del gestore dati EDI, fare riferimento a "Configurazione del meta oggetto secondario del gestore dati EDI" a pagina 96.

Analisi del documento EDI

Il gestore dati EDI analizza le informazioni nel documento EDI in base ai separatori identificati nella fase di inizializzazione. Tali separatori determinano ognuno delle differenti parti dei dati, che il gestore dati accoppia con l'attributo appropriato. Tabella 47 mostra le attività di analisi che il gestore dati effettua per i diversi oggetti di business EDI.

Tabella 47. Analisi delle attività degli oggetti di business EDI

Specifiche dell'applicazione informazioni	Attività di analisi
`type=intestazione, type=trailer`	Il gestore dati cerca la posizione dell'oggetto di business che corrisponde al segmento successivo che viene visualizzato nel documento ed analizza tale segmento per compilare l'oggetto di business secondario.
`name=` *`segment_name`* (nessuna tag `type`)	Il gestore dati assume che l'oggetto di business rappresenti un segmento ed analizza il segment corrente per compilare l'oggetto di business.
`type=loop`	Il nome del primo segmento contenuto nel loop dovrebbe essere specificato nelle informazioni specifiche dell'applicazione. Il gestore dati analizza il documento EDI per tali segmenti del loop ed aggiunge i dati all'oggetto di business.

Personalizzazione del gestore dati EDI

E' possibile personalizzare il gestore dati EDI creando uno speciale gestore nome. Il gestore dati EDI richiama il gestore nome per ottenere il nome dell'oggetto di business da creare. Il gestore dati determina quale gestore nome richiamare utilizzando il valore dell'attributo `NameHandlerClass` memorizzato nel meta oggetto del gestore dati. Il gestore nome predefinito incluso con il gestore dati EDI ricerca il nome dell'oggetto di business nel file di ricerca del gestore nome (indicato dall'attributo `NameHandlerFile` del meta oggetto). Se si desidera che il gestore nome funzioni in maniera differente, è necessario:

1. Creare un gestore nome predefinito estendendo la classe `NameHandler`.
2. Configurare il gestore dati EDI per l'utilizzo della classe del gestore nome personalizzato aggiornando il valore predefinito dell'attributo `NameHandlerClass` nel meta oggetto per il gestore dati EDI.

Per ulteriori informazioni su come creare un gestore dati personalizzato, fare riferimento a "Creazione di un gestore nome personalizzato" a pagina 194.

Capitolo 5. Gestore dati delle richieste-risposte

Il gestore dati delle richieste-risposte gestisce scenari che richiedono differenti formati di dati per le relative richieste e risposte degli oggetti di business. Il gestore dati delle richieste-risposte consente a questi due formati di essere differenti. Questo capitolo descrive come il gestore dati delle richieste-risposte elabora le informazioni e come determinare le definizioni dell'oggetto di business che devono essere elaborate dal gestore dati delle richieste-risposte. Descrive inoltre come configurare il gestore dati delle richieste-risposte.

Questo capitolo contiene le sezioni seguenti:

Nota: Il gestore dati delle richieste-risposte è uno dei gestori dati base contenuti nel file `CwDataHandler.jar`. Per ulteriori informazioni su come installare questo gestore dati e su dove memorizzare il suo codice sorgente, fare riferimento a Capitolo 2, "Installazione e configurazione del Gestore Dati", a pagina 21.

Panoramica

Il gestore dati delle richieste-risposte è un modulo di conversione dati il cui ruolo primario è fornire supporto per i dati delle richieste e delle risposte che sono in formati differenti; ossia, consente un contesto di chiamata (come ad esempio un adattatore o Server Access Interface) di richiamare due gestori dati diversi:

- Un *gestore dati di richiesta* gestisce la conversione dei dati per il componente WebSphere Business Integration Server Express che *avvia* una richiesta.
- Un *gestore dati di risposta* gestisce la conversione dei dati per il componente WebSphere Business Integration Server Express che *risponde* ad una richiesta.

Con altri gestori dati, il contesto di chiamata assume che i dati abbiano lo stesso tecnologico quando sono inviati e ricevuti dall'applicazione o dal client di accesso; perciò, il contesto di chiamata è configurato per richiamare un singolo gestore dati per gestire la conversione di entrambi gli oggetti di business di richiesta e risposta.

Tuttavia, è possibile avere un'applicazione legacy che accetti XML come un input e restituisca alcuni documenti in formato personalizzato come output. I gestori dati esistenti non possono gestire facilmente tale situazione. Tuttavia, è possibile configurare l'adattatore che comunica con questa applicazione legacy per richiamare il gestore dati di richiesta-risposta. E' possibile configurare tale gestore dati per richiamare gestori dati separati per dati in entrata e in uscita:

- Il gestore dati IBM WebSphere Business Integration Server Express per XML (gestore dati XML) per gestire gli oggetti di business di richiesta
- Un gestore dati personalizzato per gestire gli oggetti di business di risposta

Nell'elaborazione di una richiesta quando Interchange Server Express invia una richiesta a questo adattatore, l'adattatore chiama il gestore dati di richiesta-risposta, inviando l'oggetto di business di richiesta. il gestore dati di richiesta-risposta verifica la relativa configurazione per determinare che necessita di richiamare il gestore dati XML per convertire tale oggetto di business in un documento XML. Una volta che tale oggetto di business è stato convertito, il gestore dati di richiesta-risposta restituisce un documento XML all'adattatore, che successivamente lo indirizza all'applicazione legacy.

Figura 33 mostra un esempio di conversione oggetto di business stringa come effettuato dal gestore dati richiesta-risposta.

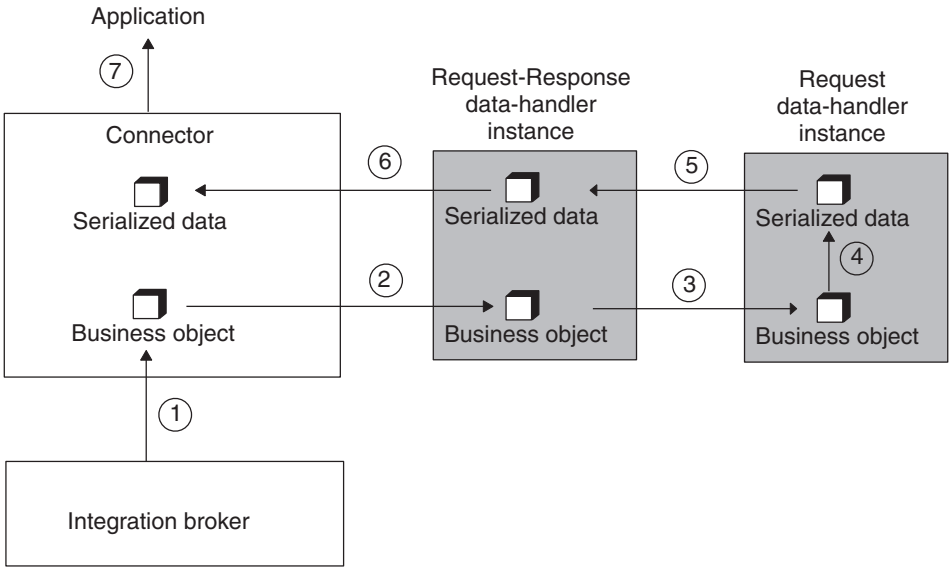

Figura 33. Conversione oggetto di business in stringa con il gestore dati richiesta-risposta

L'adattatore potrebbe ricevere successivamente una risposta dall'applicazione legacy. Tale risposta dovrebbe essere nel formato personalizzato dell'applicazione legacy. L'adattatore chiama nuovamente il gestore dati di richiesta-risposta, inviando i dati di risposta. Il gestore dati di richiesta-risposta verifica la relativa configurazione per determinare che necessita di richiamare il gestore dati personalizzato per convertire i dati di risposta in un oggetto di business. Una volta che tali dati sono stati convertiti, il gestore dati di richiesta-risposta restituisce l'oggetto di business all'adattatore, che successivamente lo indirizza a Interchange Server Express.

Il gestore dati di richiesta-risposta serve inoltre per abilitare InterChange Server Express a spedire un tipo di oggetto di business ad una porta di collaborazione e di ricevere uno o più differenti oggetti di business in cambio. Ad esempio, un client di accesso può inviare un oggetto cliente ad una collaborazione e ricevere un array di oggetti di ordini in sospeso per tale cliente in cambio.

Il gestore dati di richiesta-risposta supporta dati serializzati con `text/requestresponse` tipo MIME. Tali dati serializzati possono essere dati di testo o dati binari. Tuttavia, i meta oggetti di livello superiore predefiniti

(MO_DataHandler_Default o MO_Server_DataHandler) *non* supportano Il tipo MIME text/requestresponse. Perciò, perché un client di accesso o un connettore siano in grado di chiamare il gestore dati di richiesta-risposta, è necessario modificare il meta oggetto di livello superiore appropriato per supportare il tipo MIME text/requestresponse. Per ulteriori informazioni, consultare "Configurazione del meta oggetto di livello superiore" a pagina 132.

Questa panoramica fornisce le seguenti informazioni sul gestore dati richiesta-risposta:

- "Componenti del gestore dati richiesta-risposta"
- "Caratteristiche del gestore dati di richiesta-risposta" a pagina 124
- "Elaborazione di oggetti di business di richiesta e risposta" a pagina 128

Componenti del gestore dati richiesta-risposta

Un gestore dati può ricevere i dati serializzati in uno di questi due modi:

- Riceve i dati serializzati *e* un oggetto di business vuoto: il gestore dati immette nell'oggetto di business i dati serializzati.
- Riceve *solo* i dati serializzati: il gestore dati deve creare l'oggetto di business prima di poterlo compilare con i dati serializzati.

Il gestore dati richiesta-risposta utilizza un gestore nome per creare il nome dell'oggetto livello superiore di business che crea. Figura 34 illustra i componenti del gestore dati richiesta-risposta e le relazioni che intercorrono tra di loro.

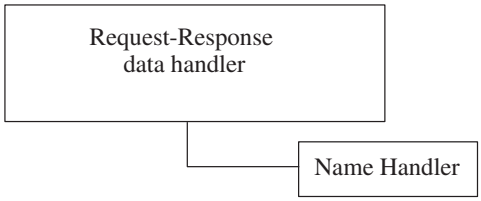

Figura 34. Componenti del gestore dati richiesta-risposta

Il gestore dati invoca un'istanza del gestore nome basata sul valore dell'attributo NameHandlerClass nel meta oggetto secondario del gestore dati di richiesta-risposta:

- Se non viene fornito nessun nome classe (l'attributo NameHandlerClass è vuoto), il gestore dati utilizza il relativo gestore nome predefinito, che aggiunge il valore del BOPrefix al nome dell'oggetto di business di livello superiore generato dal gestore dati configurato per le richieste. un errore e genera un eccezione.

  Ad esempio, se l'utente specifica l'attributo predefinito BOPrefix di REQUESTTEST e il gestore dati di richiesta genera l'oggetto di business Customer, il gestore dati di richiesta-risposta crea un oggetto di livello superiore chiamato REQUESTTEST_Customer e tenta di compilare uno dei relativi oggetti secondari con l'oggetto Customer.
- Se viene fornito un nome classe nell'attributo NameHandlerClass, il gestore dati Request-Response utilizza tale gestore nome per determinare il nome dell'oggetto di business.

  Per specificare un gestore nome personalizzato, impostare NameHandlerClass sul nome della classe del gestore nome personalizzato. Per ulteriori informazioni su come creare un gestore nome personalizzato, fare riferimento a "Personalizzazione del gestore dati di richiesta-risposta" a pagina 137.

L'attributo NameHandlerClass nella versione del meta oggetto fornito con il prodotto è vuoto. Perciò, il gestore dati di richiesta-risposta utilizza il relativo gestore nome predefinito.

Caratteristiche del gestore dati di richiesta-risposta

Il gestore dati di richiesta-risposta è utile in entrambi questi casi:

- "Supporto per l'elaborazione degli eventi"
- "Supporto per l'elaborazione della richiesta" a pagina 126

Supporto per l'elaborazione degli eventi

L'elaborazione degli eventi coinvolge la notifica di Interchange Server Express che alcuni eventi sono avvenuti, il che indica una modifica alle entità di business dell'applicazione. Nella notifica degli eventi, il contesto di chiamata di un gestore dati chiama un gestore dati: richiesto per convertire i dati serializzati in un oggetto di business (che viene poi inviato a Interchange Server Express). Tale conversione stringa in oggetto di business è effettuata dal gestore dati di richiesta, poiché tale gestore dati gestisce la conversione dal formato di richiesta (input) in un oggetto di business.

L'elaborazione degli eventi può essere sia sincrona o asincrona. Tuttavia, poiché nell'elaborazione asincrona degli eventi l'adattatore (in particolare, il componente connettore dell'adattatore) non aspetta una risposta da Interchange Server Express, allora:

- L'elaborazione asincrona degli eventi avviene quando un client di accesso notifica a WebSphere Business Integration Server Express (in particolare, alcune collaborazioni in questo server) che sono avvenuti degli eventi. Il client di accesso non attende una risposta da InterChange Server Express, né aspetta una risposta. In questo caso, Server Access Interface (in InterChange Server Express) richiama i gestori dati necessari.
- L'elaborazione asincrona degli eventi avviene quando un adattatore (in particolare, il componente connettore dell'adattatore) riceve un evento dalla relativa applicazione o tecnologia ed invia tale evento (nel formato di un oggetto di business) ad Interchange Server Express per notificare che qualcosa è avvenuto. Tuttavia, il connettore non attende una risposta da Interchange Server Express. Perciò, il gestore dati di richiesta-risposta non è utile nel caso dell'elaborazione asincrona degli eventi.

Nota: Per ulteriori informazioni sui contesti di chiamata dei gestori dati, fare riferimento a "Contesti per gestori dati richiedenti" a pagina 6.

Ad esempio, i seguenti passaggi descrivono l'elaborazione sincrona degli eventi. Descrivono come il gestore dati di richiesta-risposta consente a Server Access Interface (in InterChange Server Express) di inviare i dati in un formato (il relativo formato richiesta) a InterChange Server Express e di ricevere un formato differente di dati (il relativo formato risposta) in cambio. Ciò consente ad un client di accesso di completare uno scenario come ad esempio l'invio di un documento XML del cliente e di ricevere un documento XML contenente ordini in sospeso per tale cliente in cambio.

1. Un client di accesso invia un evento (come dati nel formato richiesta) che deve essere eseguito da una collaborazione in InterChange Server Express.
2. InterChange Server Express crea una nuova istanza del gestore dati di richiesta-risposta e passa i dati nel formato richiesta.
3. Il gestore dati di richiesta-risposta chiama il relativo gestore dati di richiesta configurato, che converte i dati in formato richiesta in un oggetto di business.

Il gestore dati di richiesta-risposta utilizza il relativo metodo `getBO()` per gestire i dati serializzati ricevuti. Richiama il gestore dati di richiesta su tali dati serializzati se una delle seguenti condizioni sono vere:

- Se il metodo riceve `getBO()` *solo* i dati serializzati come un argomento, considera tali dati come una nuova richiesta.
- Se il metodo riceve `getBO()` l'oggetto di business di livello superiore *e* i dati serializzati come un argomento, verifica gli oggetti di business secondari dell'oggetto di business di livello superiore per determinare la sua prossima azione.

  Se tutti gli oggetti di business secondari contengono un valore di CxIgnore (indicante che sono *vuoti*), il gestore dati assume che i dati serializzati rappresentano una nuova richiesta.

  Nota: Se un oggetto di business secondario in un oggetto di business di livello superiore è compilato, il gestore dati assume che tale oggetto secondario rappresenta la richiesta primaria e di conseguenza che i dati serializzati rappresentano la risposta. Perciò, crea un'istanza del gestore dati di risposta per elaborare i dati. Per ulteriori informazioni, consultare "Supporto per l'elaborazione della richiesta" a pagina 126.

In entrambi questi casi, il gestore dati di richiesta-risposta crea un'istanza del gestore dati di richiesta e utilizza tale gestore dati per convertire i dati in formato richiesta in un oggetto di business di richiesta. Al gestore dati di richiesta, il gestore dati di richiesta-risposta passa i dati in formato richiesta. Il gestore dati di richiesta restituisce il corrispondente oggetto di business di richiesta.

4. Il gestore dati di richiesta-risposta aggiunge l'oggetto di business di richiesta come un oggetto secondario dell'oggetto di business di livello superiore, come segue:
   - Se il metodo del gestore dati `getBO()` *non* ha ricevuto un oggetto di business di livello superiore, è necessario che ne crei uno nuovo e che quindi gli aggiunga l'oggetto di business di richiesta. Per denominare questo nuovo oggetto di business, il gestore dati chiama il relativo gestore nome. Per ulteriori informazioni, consultare "Componenti del gestore dati richiesta-risposta" a pagina 123.
   - Se il metodo `getBO()` del gestore dati *ha ricevuto* un oggetto di business di livello superiore, aggiunge solo l'oggetto di business di richiesta.
5. Il gestore dati di richiesta-risposta restituisce un oggetto di business di livello superiore in maniera asincrona alla collaborazione (che il client di accesso ha specificato).
6. La collaborazione riceve l'oggetto di business di livello superiore, estrae l'oggetto secondario (che contiene l'oggetto di business di richiesta) e esegue alcune elaborazioni di business.
7. La collaborazione crea un nuovo oggetto di business di risposta e lo aggiunge all'oggetto di business di livello superiore, quindi lo restituisce con successo.

   Tale oggetto di business di livello superiore è quello che la collaborazione ha ricevuto dal gestore dati di richiesta-risposta. Una volta che la collaborazione ha aggiornato tale oggetto di business, questo contiene sia l'oggetto di business di richiesta primario che l'oggetto di business di risposta appena creato.
8. InterChange Server Express passa l'oggetto di business di livello superiore modificato al gestore dati di richiesta-risposta.

9. Il gestore dati di richiesta-risposta chiama il relativo gestore dati di risposta configurato per convertire l'oggetto di business di risposta nei dati in formato risposta.

   Il gestore dati di richiesta-risposta utilizza il relativo metodo `getStringFromBO()` per gestire l'oggetto di business di livello superiore ricevuto. Se più di un oggetto di business secondario nell'oggetto di business di livello superiore è compilato, il gestore dati assume che l'oggetto di business di livello superiore contiene *sia* l'oggetto di business di richiesta primario che il relativo oggetto di business di risposta e di conseguenza che l'oggetto di business di risposta deve essere convertito. Perciò, crea un'istanza del gestore dati ti risposta per elaborare l'oggetto di business secondario definito per ultimo nell'oggetto di business di livello superiore come l'oggetto di business di risposta.

   Nota: Se solo un oggetto di business secondario nell'oggetto di business di livello superiore ricevuto da `getStringFromBO()` è compilato, il gestore dati assume che l'oggetto secondario rappresenta una nuova richiesta. Perciò, crea un'istanza del gestore dati di richiesta per convertire l'oggetto di business di richiesta in dati serializzati (in formato richiesta). Per ulteriori informazioni, consultare "Supporto per l'elaborazione della richiesta".
10. Il gestore dati di richiesta-risposta restituisce i dati in formato risposta al relativo richiedente (Server Access Framework in InterChange Server Express).
11. InterChange Server Express restituisce i dati in formato risposta (basati sull'oggetto di business di risposta) al client di accesso.

Un connettore inoltre, può effettuare l'elaborazione sincrona di un evento con il metodo `executeCollaboration()`. Generalmente, tuttavia, effettua un'elaborazione asincrona dell'evento utilizzando un meccanismo di scoperta dell'evento come ad esempio il polling.

Supporto per l'elaborazione della richiesta

L'elaborazione della richiesta coinvolge la ricezione di richieste da Interchange Server Express e l'avvio delle modifiche appropriate nelle entità di business dell'applicazione. Diversamente dall'elaborazione di un evento, che può essere avviata sia da un client di accesso (in maniera sincrona) che da un connettore (in maniera asincrona), l'elaborazione della richiesta viene avviata da InterChange Server Express e coinvolge solo le comunicazioni con i connettori (non con i client di accesso).

Nella elaborazione della richiesta, il contesto di chiamata di un gestore dati chiama un gestore dati per convertire i dati serializzati in un oggetto di business (che viene quindi indirizzato a InterChange Server Express). Tale conversione stringa in oggetto di business

Ad esempio, i seguenti passaggi descrivono l'elaborazione della richiesta che coinvolge InterChange Server Express ed un adattatore tecnologico. Questo adattatore tecnologico è configurato per utilizzare il gestore dati di richiesta-risposta per gestire dati di richiesta e di risposta. In tale caso, i dati di richiesta sono in un formato e i dati di risposta in un altro.

1. La collaborazione crea una nuova istanza dell'oggetto di business di livello superiore e aggiunge l'oggetto di business di richiesta come un elemento secondario.

2. La collaborazione invia una richiesta (nel formato dell'oggetto di business di livello superiore) al connettore tecnologico, che in seguito passa l'oggetto di business di livello superiore al gestore dati di richiesta-risposta.
3. Il gestore dati di richiesta-risposta chiama il relativo gestore dati di richiesta configurato per convertire l'oggetto di business di richiesta nei dati in formato richiesta.

   Il gestore dati di richiesta-risposta utilizza il relativo metodo `getStringFromBO()` per gestire l'oggetto di business di livello superiore ricevuto. Se è compilato solamente un oggetto di business secondario nell'oggetto di business di livello superiore, il gestore dati di richiesta-risposta assume che l'oggetto secondario rappresenta una nuova richiesta. Perciò, crea un'istanza del gestore dati di richiesta per convertire l'oggetto di business di richiesta in dati serializzati (in formato richiesta).

   Nota: Se è compilato più di un oggetto di business secondario nell'oggetto di business di livello superiore ricevuto da `getStringFromBO()`, il gestore dati assume che l'oggetto di business di livello superiore contiene *sia* l'oggetto di business primario di richiesta sia il relativo oggetto di business di risposta e di conseguenza che è necessario convertire l'oggetto di business di risposta. Perciò, crea un'istanza del gestore dati ti risposta per elaborare l'oggetto di business secondario definito per ultimo nell'oggetto di business di livello superiore come l'oggetto di business di risposta. Per ulteriori informazioni, consultare "Supporto per l'elaborazione degli eventi" a pagina 124.
4. Il connettore tecnologico invia i dati in formato richiesta all'applicazione.
5. L'applicazione effettua determinate operazioni e restituisce i dati in formato risposta al connettore tecnologico.
6. Il connettore tecnologico passa sia l'oggetto di business primario di livello superiore che i dati in formato risposta al gestore dati di richiesta-risposta.
7. Il gestore dati di richiesta-risposta chiama il relativo gestore dati di risposta configurato per convertire i dati in formato risposta in un oggetto di business.

   Il gestore dati di richiesta-risposta utilizza il relativo metodo `getBO()` per gestire l'oggetto di business di livello superiore e i dati serializzati ricevuti. Se è compilato uno qualsiasi degli oggetti di business secondari nell'oggetto di business di livello superiore, il gestore dati di richiesta-risposta assume che l'oggetto secondario rappresenta la richiesta primaria e di conseguenza che i dati serializzati ricevuti sono nel formato di risposta. Perciò, crea un'istanza del gestore dati di risposta per convertire i dati serializzati in un oggetto di business di risposta.

   Nota: Se tutti gli oggetti di business secondari nell'oggetto di business di livello superiore ricevuto da `getBO()` contengono un valore di CxIgnore (che indica che sono *vuoti*), il gestore dati assume che i dati serializzati rappresentano una nuova richiesta e crea un'istanza del gestore dati di richiesta per elaborare i dati. Per ulteriori informazioni, consultare "Supporto per l'elaborazione degli eventi" a pagina 124.
8. Il gestore dati di richiesta-risposta aggiunge tale oggetto di business di risposta come secondario dell'oggetto di business di livello superiore e quindi restituisce questo oggetto di business di livello superiore al relativo richiedente (il connettore tecnologico).
9. Il connettore tecnologico restituisce l'oggetto di business di livello superiore aggiornato alla collaborazione.

10. La collaborazione riceve l'oggetto di business di livello superiore e immagazzina il contenuto del relativo oggetto di business di risposta nell'elaborazione di business.

Elaborazione di oggetti di business di richiesta e risposta

L'utilizzo del gestore dati di richiesta-risposta per convertire un oggetto di business di richiesta nel relativo formato di richiesta appropriato o per convertire i dati in un formato di risposta in un oggetto di business di risposta richiede che si verifichino i passaggi descritti in Tabella 48.

Tabella 48. Utilizzo del gestore dati di richiesta-risposta

Passo	Per ulteriori informazioni
1. E' necessario che le definizioni dell'oggetto di business che descrivono la struttura dell'oggetto di business esistano e che siano disponibili per il gestore dati di richiesta-risposta (e per i relativi gestori dati del componente) quando li esegue.	"Requisiti per le definizioni oggetto di business"
2. E' necessario che il gestore dati di richiesta-risposta sia configurato per il proprio ambiente.	"Configurazione del gestore dati di richiesta-risposta" a pagina 132
3. E' necessario che il gestore dati di richiesta-risposta sia chiamato da un contesto di chiamata (connettore o client di accesso) per effettuare le operazioni di dati appropriate:	
a) Operazioni di dati: Riceve la richiesta dal componente che avvia tale richiesta e la converte nel formato appropriato.	"Conversione degli oggetti di business con il gestore dati di richiesta" a pagina 136
b) Operazioni di dati: Riceve la risposta dal componente che risponde alla richiesta e la converte nel formato appropriato.	"Conversione degli oggetti di business con il gestore dati di risposta" a pagina 136

Requisiti per le definizioni oggetto di business

Per utilizzare il gestore dati di richiesta-risposta, è necessario creare o modificare le definizioni dell'oggetto di business in modo che forniscano la struttura richiesta dal gestore dati. Tuttavia, diversamente dagli altri gestori dati, non è necessario modificare le definizioni dell'oggetto di business in modo da contenere i metadati. Questa sezione fornisce le informazioni necessarie per creare le definizioni dell'oggetto di business per funzionare con il gestore dati di richiesta-risposta. In particolare, fornisce le seguenti informazioni:

- "Comprensione della struttura dell'oggetto di business di richiesta-risposta"
- "Creazione delle definizioni dell'oggetto di business per il gestore dati di richiesta-risposta" a pagina 131

Comprensione della struttura dell'oggetto di business di richiesta-risposta

Il gestore dati di richiesta-risposta utilizza le definizioni dell'oggetto di business quando riceve un oggetto di business o invia oggetti di business al relativo gestore dati di richiesta di risposta. Il gestore dati di richiesta-risposta dispone requisiti specifici sulla struttura degli oggetti di business che può elaborare. E' necessario che gli oggetti di business passati al gestore dati contengano un oggetto secondario

di richiesta e uno o più oggetti secondari di risposta. E' necessario che tali oggetti secondari corrispondano ai requisiti del gestore dati che li elaborerà.

Nota: Il gestore dati di richiesta-risposta *non* richiede informazioni specifiche dell'applicazione nelle definizioni dell'oggetto di business o nei relativi attributi.

Figura 35 mostra la struttura dell'oggetto di business che rappresenta un oggetto di business di richiesta-risposta.

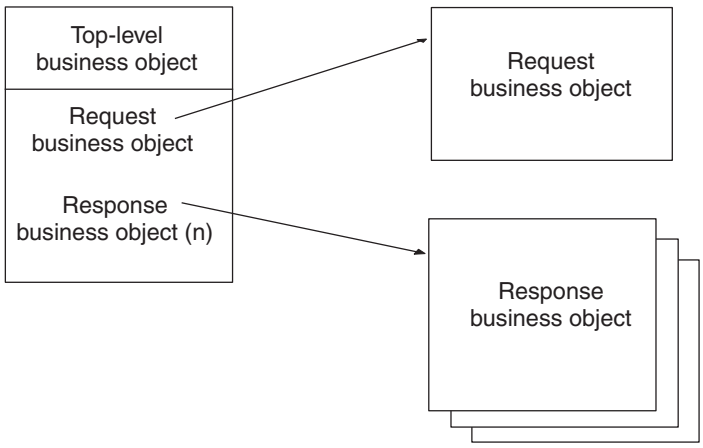

Figura 35. Struttura dell'oggetto di business per un oggetto di business di richiesta-risposta

Per assicurarsi che le definizioni dell'oggetto di business corrispondono ai requisiti del gestore dati di richiesta-risposta, utilizzare le indicazioni fornite per ciascuno dei seguenti oggetti di business:

- "Oggetto di business di livello superiore"
- "Oggetto di business di richiesta" a pagina 130
- "Oggetto di business di risposta" a pagina 131

Oggetto di business di livello superiore

Il gestore dati di richiesta-risposta prevede un oggetto di business di livello superiore per contenere le informazioni ricevute o inviate dal relativo contesto di chiamata. Tabella 49 descrive come il gestore dati di richiesta-risposta interpreta le proprietà di un oggetto di business e descrive come impostare le proprietà quando si modifica un oggetto di business per l'utilizzo con il gestore dati di richiesta-risposta.

Tabella 49. Proprietà per le definizioni dell'oggetto di business di livello superiore

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. Si raccomanda che tali definizioni degli oggetti di business inizino con un prefisso standard. Il nome dell'oggetto di business di livello superiore dipende dallo standard del messaggio, come mostrato di seguito: `BusObj Prefix + Response Business Object`
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.

Tabella 49. Proprietà per le definizioni dell'oggetto di business di livello superiore (Continua)

Nome proprietà	Descrizione
Informazioni specifiche dell'applicazione	Non è richiesta nessuna informazione specifica dell'applicazione.

E' necessario che questo oggetto di business di livello superiore contenga i seguenti attributi:

- Un attributo a cardinalità singola per contenere un oggetto di business di richiesta singolo

  E' necessario che tale oggetto di business di richiesta corrisponda ai requisiti dell'oggetto di business richiesti dal gestore dati di richiesta.
- Un attributo a cardinalità singola per contenere gli oggetti di business di risposta

  E' necessario che tale oggetto di business di risposta corrisponda ai requisiti dell'oggetto di business richiesti dal gestore dati di risposta.

  Note:

  1. Se l'applicazione di destinazione può restituire *più di un* tipo di oggetto di business di risposta, è necessario che l'oggetto di business di livello superiore contenga un oggetto di business secondario per ogni tipo di oggetto di business di risposta.

     Ad esempio, se l'applicazione di destinazione può restituire sia un documento XML Customer che un documento XML OrderUpdate, è necessario che la definizione dell'oggetto di business includa due attributi, uno per contenere la definizione dell'oggetto di business object che rappresenta il documento XML Customer e il secondo per contenere la definizione dell'oggetto di business che contiene il documento XML OrderUpdate. Quando riceve l'oggetto di business di risposta, il gestore dati compila l'attributo appropriato.
  2. Il gestore dati fallisce l'elaborazione se riceve un oggetto di business che *non* corrisponde alle attese o se altri gestori dati coinvolti non riescono a convertire gli oggetti di business o i documenti passati.

Oggetto di business di richiesta

Per contenere le informazioni di richiesta per il gestore dati di richiesta, il gestore dati di richiesta-risposta prevede un oggetto di business di richiesta come il primo attributo dell'oggetto di business di livello superiore. Tale attributo dovrebbe essere a cardinalità singola. Tabella 50 descrive come il gestore dati di richiesta-risposta interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare tali proprietà durante la modifica dell'oggetto di business per l'utilizzo con il gestore dati di richiesta-risposta.

Tabella 50. Proprietà per le definizioni dell'oggetto di business di richiesta

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. E' necessario che tale nome corrisponda al nome della definizione dell'oggetto di business che il gestore dati di richiesta gestisce.
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Dipende dal particolare gestore dati di richiesta che viene utilizzato.

Nota: Per ulteriori informazioni sul formato dell'oggetto di business di richiesta, fare riferimento alla documentazione per il gestore dati che agisce come il gestore dati di richiesta.

Ad esempio, se si specifica `RequestDataHandlerMimeTypecome` `text/xml`, l'oggetto secondario definito come il relativo oggetto di business di richiesta deve essere compatibile con il gestore dati XML.

Oggetto di business di risposta

Per contenere le informazioni di risposta per il gestore dati di risposta, il gestore dati di richiesta-risposta prevede un oggetto di business di risposta negli attributi che inizi con il secondo attributo dell'oggetto di business di livello superiore. Tale attributo dovrebbe essere a cardinalità singola. Se il gestore dati di risposta restituisce più di un tipo di oggetto di business, l'oggetto di business di livello superiore contiene un attributo per ogni tipo possibile. Tabella 51 descrive come il gestore dati di richiesta-risposta interpreta le proprietà di tale definizione dell'oggetto di business e descrive come impostare tali proprietà durante la modifica dell'oggetto di business per l'utilizzo con il gestore dati di richiesta-risposta.

Tabella 51. Proprietà per le definizioni dell'oggetto di business di risposta

Nome proprietà	Descrizione
Name	E' necessario che ogni definizione dell'oggetto di business abbia un nome univoco. E' necessario che tale nome corrisponda al nome della definizione dell'oggetto di business che il gestore dati di risposta gestisce.
Versione	Una costante rappresenta la versione corrente della definizione dell'oggetto di business. Il valore corrente è 1.0.0.
Informazioni specifiche sull'applicazione	Dipende dal particolare gestore dati di richiesta che viene utilizzato.

Nota: Per ulteriori informazioni sul formato dell'oggetto di business di richiesta, fare riferimento alla documentazione per il gestore dati che agisce come il gestore dati di richiesta.

Ad esempio, se si specifica `ResponseDataHandlerMimeType` come `text/abc`, l'oggetto secondario definito come il relativo oggetto di business di richiesta deve essere compatibile con un gestore dati personalizzato che può gestire il tipo MIME `abc`.

Creazione delle definizioni dell'oggetto di business per il gestore dati di richiesta-risposta

Tale sezione descrive come creare definizioni dell'oggetto di business per rappresentare la struttura che il gestore dati di richiesta-risposta prevede. Utilizzare Business Object Designer Express per aggiungere o eliminare attributi dalla definizione dell'oggetto di business oltre a modificare le proprietà dell'attributo, se necessario.

Come descritto in "Comprensione della struttura dell'oggetto di business di richiesta-risposta" a pagina 128, il gestore dati di richiesta-risposta richiede la creazione delle seguenti definizioni dell'oggetto di business:

- "Creazione della definizione dell'oggetto di business di livello superiore" a pagina 132
- "Creazione di altre definizioni dell'oggetto di business" a pagina 132

Creazione della definizione dell'oggetto di business di livello superiore

Per creare la definizione dell'oggetto di business di livello superiore per il gestore dati di richiesta-risposta, è necessario creare manualmente una definizione dell'oggetto di business utilizzando Business Object Designer Express:

1. Creare la definizione dell'oggetto di business di livello superiore.

   Per ulteriori informazioni sulla struttura di tale oggetto di business di livello superiore, fare riferimento a "Oggetto di business di livello superiore" a pagina 129.
2. Creare gli oggetti di business secondari per l'oggetto di business di livello superiore. Nell'oggetto di business di livello superiore, creare un attributo dell'oggetto secondario per gli oggetti di business mostrati in Tabella 52.

Tabella 52. Oggetti di business per il gestore dati di richiesta-risposta

Attributo	Note	Oggetto business
Oggetto di business di richiesta	Contiene le informazioni riguardanti la richiesta.	"Oggetto di business di richiesta" a pagina 130
Oggetto di business di risposta	Contiene le informazioni riguardanti la risposta alla richiesta	"Oggetto di business di risposta" a pagina 131

Creazione di altre definizioni dell'oggetto di business

Per creare le richiesta e definizioni di un oggetto di business di risposta, è possibile utilizzare uno dei seguenti modi:

- E' possibile utilizzare ODA (Object Discovery Agent) per esportare i dati serializzati come una definizione dell'oggetto di business. Gli ODA esistono per numerosi formati di dati, inclusi i documenti XML.

  Per ulteriori informazioni su ODA XML, fare riferimento a "Utilizzare l'ODA XML", a pagina 217. Per informazioni su altri ODA, fare riferimento a guida dell'adattatore corrispondente.
- E' possibile creare manualmente una definizione dell'oggetto di business per i dati utilizzando Business Object Designer Express.

  Per ulteriori informazioni, consultare "Creazione della definizione dell'oggetto di business di livello superiore"

Configurazione del gestore dati di richiesta-risposta

Il gestore dati di richiesta-risposta recupera le relative proprietà di configurazione da una gerarchia di meta oggetti, come segue:

- Il meta oggetto principale consente ad un connettore o a Server Access Interface di istanziare un gestore dati in base al tipo MIME del documento.
- Il meta oggetto secondario contiene tutte le informazioni necessarie per elaborare i dati nel documento, inclusi il nome classe del gestore dati, il prefisso dell'oggetto di business da creare ed altri.

Configurazione del meta oggetto di livello superiore

Il tipo MIME contenuto nel meta oggetto principale indica quali tipi MIME sono supportati e quali gestori dati forniscono tale supporto. Nessuno dei meta oggetti di livello superiore forniti include una voce per il gestore dati di richiesta-risposta. Per abilitare il proprio connettore o un client di accesso ad utilizzare il gestore dati di richiesta-risposta, è necessario aggiungere un attributo per il tipo MIME

text/requestresponse del meta oggetto di livello superiore MO_Server_DataHandler o MO_DataHandler_Default. E' necessario che tale attributo sia del tipo MO_DataHandler_DefaultRequestResponseConfig.

Il seguente frammento di una definizione dell'oggetto di business mostra la definizione per l'attributo text/requestresponse:

```
[Attribute]
Name = text.requestresponse
Type = MO_DataHandler_DefaultRequestResponseConfig
ContainedObjectVersion = 1.0.0
Relationship = Containment
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
```

Nota: Per ulteriori informazioni sui meta oggetti di livello superiore e su come modificarli, consultare "meta oggetti principali" a pagina 22.

Configurazione del meta oggetto secondario

Per configurare un gestore dati di richiesta-risposta, è necessario assicurarsi che le relative informazioni di configurazione siano fornite nel meta oggetto secondario di richiesta-risposta.

Nota: Per configurare un gestore dati di richiesta-risposta, è necessario inoltre creare o modificare le definizioni dell'oggetto di business in modo che supportino il gestore dati. Per ulteriori informazioni, consultare "Requisiti per le definizioni oggetto di business" a pagina 128.

Per il gestore dati di richiesta-risposta, IBM fornisce il meta oggetto secondario predefinito MO_DataHandler_DefaultRequestResponseConfig. Ogni attributo in questo meta oggetto definisce una proprietà di configurazione per il gestore dati richiesta-risposta. Tabella 53 descrive gli attributi in questo meta oggetto secondario.

Tabella 53. Attributi meta oggetto secondario per il gestore dati di richiesta-risposta

Nome attributo	Descrizione	Valore predefinito fornito
BOPrefix	Il prefisso utilizzato dalla classe predefinita NameHandler per creare il nome dell'oggetto di business di livello superiore. Il valore predefinito deve essere modificato per corrispondere al nome della definizione dell'oggetto di business associato. Il valore dell'attributo è sensibile al maiuscolo/minuscolo.	REQUESTTEST
ClassName	Nome della classe gestore dati da caricare per l'utilizzo con il tipo MIME specificato. E' necessario che il meta oggetto del gestore dati di livello superiore contenga un attributo il cui nome corrisponda al tipo MIME specificato e il cui tipo sia il meta oggetto secondario di richiesta -risposta (descritto da questa tabella).	com.crossworlds.DataHandlers.text.requestresponse
DefaultVerb	Nome del verbo da impostare nell'oggetto di business durante la conversione da un documento di richiesta in un oggetto di business. Se non esiste nessun valore per tale documento, il gestore dati di richiesta-risposta non include un verbo nell'oggetto di business	Create

Tabella 53. Attributi meta oggetto secondario per il gestore dati di richiesta-risposta (Continua)

Nome attributo	Descrizione	Valore predefinito fornito
NameHandlerClass	Nome della classe del gestore nome da utilizzare per determinare il nome dell'oggetto di business di livello superiore dal contenuto di un documento di richiesta. Modificare il valore predefinito di questo attributo se si intende creare il proprio gestore dati personalizzato. Per ulteriori informazioni, consultare "Creazione di un gestore nome personalizzato XML" a pagina 89.	com. crossworlds. DataHandlers.xml. TopElementNameHandler
RequestDataHandlerMimeType	Il tipo MIME delle richieste elaborate da tale gestore dati. Il gestore dati di richiesta-risposta utilizza tale tipo MIME per determinare il gestore dati da istanziare per l'elaborazione di ogni oggetto di business di richiesta or documenti.	text/xml
ResponseDataHandlerMimeType	Il tipo MIME delle risposte elaborate da tale gestore dati. Il gestore dati di richiesta risposta utilizza tale tipo MIME per determinare il gestore dati da istanziare per l'elaborazione di ogni oggetto di business o documento.	text/xml
ObjectEventId	Placeholder non viene utilizzato dal gestore dati ma è richiesto dal sistema di integrazione business.	*Nessuno*

Il "Valore predefinito fornito" colonna in Tabella 53 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente nell'oggetto di business fornito. E' necessario verificare il proprio ambiente e impostare le proprietà del Valore predefinito di tali attributi con i valori appropriati. E' necessario assicurarsi che almeno gli attributi ClassName e BOPrefix abbiano valori predefiniti.

Per creare il meta oggetto secondario MO_DataHandler_DefaultRequestResponseConfig, utilizzare Business Object Designer Express per creare una definizione dell'oggetto di business con il seguente formato:

```
[BusinessObjectDefinition]
Name = MO_DataHandler_DefaultRequestResponseConfig
Version = 1.0.0

   [Attribute]
   Name = ClassName
   Type = String
   Cardinality = 1
   MaxLength = 255
   IsKey = false
   IsForeignKey = false
   IsRequired = false
   DefaultValue = com.crossworlds.DataHandlers.text.requestresponse
   IsRequiredServerBound = false
   [End]

   [Attribute]
   Name = NameHandlerClass
   Type = String
   MaxLength = 255
   IsKey = false
   IsForeignKey = false
   IsRequired = false
   IsRequiredServerBound = false
   [End]

   [Attribute]
   Name = RequestDataHandlerMimeType
   Type = String
   MaxLength = 255
   IsKey = false
```

```
   IsForeignKey = false
   IsRequired = false
   DefaultValue = text/xml
   IsRequiredServerBound = false
   [End]
   [Attribute]
   Name = ResponseDataHandlerMimeType
   Type = String
   MaxLength = 255
   IsKey = false
   IsForeignKey = false
   IsRequired = false
   DefaultValue = text/xml
   IsRequiredServerBound = false
   [End]
   [Attribute]
   Name = BOPrefix
   Type = String
   MaxLength = 255
   IsKey = false
   IsForeignKey = false
   IsRequired = false
   DefaultValue = Wrapper
   IsRequiredServerBound = false
   [End]
   [Attribute]
   Name = DummyKey
   Type = String
   MaxLength = 1
   IsKey = true
   IsForeignKey = false
   IsRequired = false
   DefaultValue = 1
   IsRequiredServerBound = false
   [End]
   [Attribute]
   Name = ObjectEventId
   Type = String
   Cardinality = 1
   MaxLength = 255
   IsKey = false
   IsForeignKey = false
   IsRequired = false
   IsRequiredServerBound = false
   [End]
   [Istruzione]
   Name = Create
   [End]
   [Istruzione]
   Name = Delete
   [End]
   [Istruzione]
   Name = Retrieve
   [End]
   [Istruzione]
   Name = Update
   [End]
[End]
```

Fare riferimento a “Configurazione dei gestori dati” a pagina 21 per le informazioni su dove posizionare tale file del meta oggetto secondario.

Conversione degli oggetti di business con il gestore dati di richiesta

Il *gestore dati di richiesta* gestisce la conversione di dati per il componente WebSphere Business Integration Server Express che avvia una richiesta:

- Nell'elaborazione della richiesta, Interchange Server Express avvia la richiesta, nella forma di un oggetto di business di richiesta inviato ad un connettore.

  InterChange Server Express
  Con InterChange Server Express, la collaborazione crea l'oggetto di business di richiesta e lo invia al connettore appropriato per l'elaborazione della richiesta.

  Perciò, il gestore dati di richiesta effettua una conversione dell'oggetto di business in stringa sulla richiesta, convertendo l'oggetto di business di richiesta in dati serializzati. Tali dati serializzati sono nel *formato di richiesta*, che è il formato che l'applicazione del connettore (o il client di accesso) accetta come input.
- Nell'elaborazione degli eventi, il contesto di chiamata (client di accesso o connettore) avvia la richiesta, nel formato di dati serializzati inviati a Interchange Server Express.

  InterChange Server Express
  Con InterChange Server Express, è possibile utilizzare un client di accesso per avviare l'elaborazione sincrona dell'evento. Per ulteriori informazioni, consultare "Supporto per l'elaborazione degli eventi" a pagina 124.

  Perciò, il gestore dati di richiesta effettua una conversione di stringa in oggetto di business sulla richiesta, convertendo i dati serializzati in un oggetto di business di richiesta. Tali dati serializzati sono nel formato di richiesta, che è il formato che il client di accesso o l'applicazione del connettore generano come output.

Il gestore dati di richiesta-risposta determina quale gestore dati richiamare come relativo gestore dati di richiesta in base alla proprietà `RequestDataHandlerMimeType` nel meta oggetto secondario. La proprietà `RequestDataHandlerMimeType` contiene il tipo MIME che il gestore dati di richiesta supporta. Se `RequestDataHandlerMimeType` *non* è stata inizializzata, il gestore dati di richiesta-risposta registra un errore e genera un'eccezione. Perciò, è *necessario* inizializzare la proprietà `RequestDataHandlerMimeType`.

Conversione degli oggetti di business con il gestore dati di risposta

Il *gestore dati di risposta* gestisce la conversione di dati per il componente WebSphere Business Integration Server Express che risponde ad una richiesta:

- Nella elaborazione di richiesta, l'applicazione del connettore risponde alla richiesta, nella forma di dati serializzati inviati al connettore, che in seguito lo indirizza a InterChange Server Express.

InterChange Server Express
Con InterChange Server Express, la collaborazione riceve l'oggetto di business di risposta dal connettore.

Perciò, il gestore dati di risposta effettua una conversione stringa in oggetto di business sulla risposta, convertendo i dati serializzati in un oggetto di business di risposta. Tali dati serializzati sono nel *formato di risposta*, che è il formato che l'applicazione del connettore (o il client di accesso) genera come output.

- Nell'elaborazione dell'evento, Interchange Server Express risponde alla richiesta, nel formato di un oggetto di business di risposta inviato al contesto di chiamata (client di accesso o connettore).

  Perciò, il gestore dati di risposta effettua una conversione oggetto di business in stringa sulla risposta, convertendo l'oggetto di business di risposta in dati serializzati. Tali dati serializzati sono nel formato di risposta, che è il formato che il client di accesso o l'applicazione del connettore accettano come input.

Il gestore dati di richiesta-risposta determina quale gestore dati richiamare come relativo gestore dati di risposta in base alla proprietà `ResponseDataHandlerMimeType` nel meta oggetto secondario. La proprietà `ResponseDataHandlerMimeType` contiene il tipo MIME che il gestore dati di risposta supporta. Se `ResponseDataHandlerMimeType` *non* è stata inizializzata, il gestore dati di richiesta-risposta registra un errore e genera un'eccezione. Perciò, è *necessario* inizializzare la proprietà `ResponseDataHandlerMimeType`.

Gestione degli errori

Il gestore dati di richiesta-risposta emette un'eccezione se non può elaborare una richiesta. Il gestore dati di richiesta-risposta fornisce un messaggio di eccezione che descrive sia cosa il gestore dati tentava di fare quando è avvenuto l'errore sia ogni messaggio restituito da un componente coinvolto nella transazione (ad esempio un altro gestore dati o il JCDK). Tali eccezioni sono inviate (forse non come eccezione ma come un'altra struttura di dati) al componente che ha richiamato inizialmente il gestore dati.

Il gestore dati di richiesta-risposta registra un errore e traccia i messaggi utilizzando i servizi forniti dal gestore dati framework. I messaggi di errore e di avviso sono registrati nei log di InterChange Server Express. Il gestore dati non registra nessun messaggio nella console Java.

Personalizzazione del gestore dati di richiesta-risposta

E' possibile personalizzare il gestore dati di richiesta-risposta creando uno speciale gestore nome. Il gestore dati di richiesta-risposta richiama il gestore nome per ottenere il nome dell'oggetto di business da creare. Il gestore dati determina quale gestore nome richiamare utilizzando il valore dell'attributo `NameHandlerClass` memorizzato nel meta oggetto del gestore dati. Il gestore nome predefinito incluso con il gestore dati di richiesta-risposta aggiunge il valore dell'attributo `BOPrefix` al nome dell'oggetto di business che il gestore dati di risposta restituisce. Se si desidera che il gestore nome funzioni in maniera differente, è necessario:

1. Creare un gestore nome predefinito estendendo la classe `NameHandler`.

2. Configurare il gestore dati di richiesta-risposta per l'utilizzo della classe del gestore nome personalizzato aggiornando il valore predefinito dell'attributo `NameHandlerClass` nel meta oggetto per il gestore dati Request-Response.

Per ulteriori informazioni su come creare un gestore dati personalizzato, fare riferimento a "Creazione di un gestore nome personalizzato" a pagina 194.

Capitolo 6. gestore dati FixedWidth

Il gestore dati FixedWidth converte gli oggetti di business in stringhe e flussi a larghezza fissa e viceversa. Questo capitolo descrive come il gestore dati FixedWidth elabora documenti a larghezza fissa e come definire gli oggetti di business che devono essere elaborati dal gestore dati. E' possibile utilizzare questa guida per implementare gli oggetti di business che si adattano ai requisiti del gestore dati FixedWidth. Questo capitolo descrive, inoltre, come configurare il gestore dati FixedWidth. Questo capitolo contiene le sezioni seguenti:

Nota: Il gestore dati FixedWidth è uno dei gestori dati base contenuti nel file `CwDataHandler.jar`. Per ulteriori informazioni su come installare questo gestore dati e su dove memorizzare il suo codice sorgente, fare riferimento a Capitolo 2, "Installazione e configurazione del Gestore Dati", a pagina 21.

Panoramica

Il gestore dati FixedWidth è un modulo di conversione dati il cui ruolo primario è convertire oggetti di business in stringhe o flussi che hanno un formato di campi a larghezza fissa e viceversa. Una stringa o un flusso a larghezza fissa è un dato serializzato con `testo/fixedwidth` tipo MIME. Il gestore dati analizza dati testo utilizzando campi a larghezza fissa. La lunghezza del campo viene specificata dalla dalla proprietà MaxLength di ciascun attributo oggetto di business. Il valore di tale proprietà è memorizzato nella definizione dell'oggetto di business.

Il gestore dati FixedWidth supporta inoltre grandi oggetti di business, che sono definiti come oggetti di business tra 10 e 100 megabyte in grandezza.

Il meta oggetto connettore di livello superiore predefinito (`MO_DataHandler_Default`) supporta il tipo MIME `text/fixedwidth`. Perciò, un connettore configurato per supportare il meta oggetto gestore dati `MO_DataHandler_Default` può richiedere il gestore dati FixedWidth. Perché un client di accesso sia in grado di richiedere questo gestore dati utilizzando Interchange Server Express, è necessario modificare il meta oggetto `MO_Server_DataHandler` affinché supporti il tipo MIME `text/fixedwidth`. Per ulteriori informazioni, consultare "Modifica del meta-oggeto di livello superiore" a pagina 197.

Caratteristiche del gestore dati FixedWidth

Il gestore dati FixedWidth utilizza il valore della proprietà MaxLength degli attributi in una definizione dell'oggetto di business per determinare come leggere e scrivere i dati. `MaxLength` definisce il numero massimo di caratteri del valore attributo, inclusa la variabile padding per consentire il testo giustificato a destra o sinistra.

E' possibile configurare un carattere pad che rappresenta gli spazi da aggiungere o rimuovere dai dati per l'allineamento. I caratteri pad vengono aggiunti quando si convertono oggetti di business in stringhe e rimossi quando si convertono le

stringhe in oggetti di business. E' possibile, inoltre, configurare l'allineamento del valore attributo oggetto di business per giustificarlo a sinistra o a destra. In questo modo è possibile per i dati del valore attributo mantenere caratteri significativi. Tabella 54 descrive i valori per l'attributo del meta oggetto `Allineamento`.

Tabella 54. Valori dell'attributo meta oggetto Allineamento

Valore	Descrizione
SINISTRA	Elimina il lato sinistro e allinea il lato destro.
DESTRA	Elimina il lato destro e allinea il lato sinistro.
ENTRAMBI	Durante la conversione delle stringhe in oggetti di business, elimina entrambi i lati. Durante la conversione di oggetti di business in stringhe, allinea il lato destro con caratteri pad.

Inoltre, con l'attributo del meta oggetto `Truncation`, è possibile configurare il gestore dati FixedWidth data per ridurre i valori dell'oggetto di business ai loro `MaxLength` o generare un errore se un valore dell'attributo è una stringa più lunga di `MaxLength`. E' inoltre possibile impostare la misura della lunghezza da utilizzare per il nome oggetto di business, il verbo e il conteggio oggetto di business per oggetti secondari di cardinalità 1-n.

Utilizzare Business Object Designer Express per impostare il valore della proprietà `MaxLength` dell'attributo stringa. Per modificare il valore della proprietà MaxLength di altri tipi (come ad esempio Intero, Doppio e così via), è necessario salvare la definizione dell'oggetto di business in un file, modificarlo manualmente e quindi importare la definizione modificata nel sistema di integrazione business.

Elaborazione di un oggetto di business e di una stringa FixedWidth

Il gestore dati FixedWidth effettua le operazioni elencate in Tabella 55.

Tabella 55. Operazioni dati per il gestore dati FixedWidth

operazione gestore dati	Per ulteriori informazioni
Riceve un oggetto di business dal richiedente, converte l'oggetto di business in una stringa o flusso FixedWidth e trasferisce i dati FixedWidth al richiedente.	"Conversione oggetti di business in documenti FixedWidth" a pagina 144
Riceve una stringa o un flusso dal richiedente, crea un oggetto di business e invia di nuovo l'oggetto di business al richiedente.	"Conversione documenti FixedWidth in oggetti di business" a pagina 145

Configurazione del gestore dati FixedWidth

Per configurare il gestore dati FixedWidth, procedere nel modo seguente:

- Immettere i valori appropriati per gli attributi del meta oggetto secondario FixedWidth.
- Creare o modificare le definizioni oggetto di business in modo che supportino il gestore dati.

Ognuno di questi passaggi è descritto con maggiori dettagli nelle sezioni seguenti.

Configurazione del meta oggetto secondario FixedWidth

Per configurare un gestore dati FixedWidth, è necessario assicurarsi che le sue informazioni di configurazione siano fornite nel meta oggetto secondario FixedWidth. Per il gestore dati FixedWidth, IBM fornisce il meta oggetto

secondario MO_DataHandler_DefaultFixedWidthConfig. Ogni attributo in questo meta oggetto definisce una proprietà di configurazione per il gestore dati FixedWidth. Tabella 56 descrive gli attributi per questo meta oggetto secondario.

Tabella 56. Attributi meta oggetto secondario per il gestore dati FixedWidth

Nome attributo meta oggetto	Significato	Valore predefinito fornito
ClassName	Nome della classe del gestore dati da caricare per l'utilizzo con il tipo MIME corrispondente al nome dell'attributo nel meta oggetto superiore del gestore dati. Tale attributo ha il meta oggetto secondario FixedWidth come il suo tipo.	com.crossworlds.DataHandlers.text.fixedwidth
Allineamento	Aggiunge o rimuove l'attributo CaratterePad. Per l'elaborazione di un evento, i caratteri pad sono eliminati. Per l'elaborazione di una richiesta, i caratteri pad sono aggiunti. I valori possibili sono ENTRAMBI, SINISTRA e DESTRA. Ad esempio, l'allineamento "SINISTRA" significa che il valore degli attributi oggetto di business si muove all'estrema sinistra dello spazio per questo valore attributo. allineamento"ENTRAMBI" per la notificazione di un evento significa che, i caratteri pad sono eliminati sia dal lato sinistro che dal destro. Allineamento"DESTRA" per una richiesta di elaborazione significa che, il lato destro è allineato con caratteri pad.	ENTRAMBI
BOCountSize	Specifica lo spazio assegnato per il numero totale di oggetti di business che devono essere elaborati.	3
BONameSize	Specifica lo spazio assegnato per il nome dell'oggetto di business.	50
BOVerbSize	Specifica lo spazio assegnato per il verbo.	20
CxBlank	Durante la conversione da un oggetto di business, il gestore dati FixedWidth scrive il valore configurato per la proprietà Valore predefinito dell'attributo meta oggetto CxBlank al documento di larghezza fissa qualora incontri un attributo oggetto di business il cui valore sia CxBlank. Durante la conversione di un oggetto di business, il gestore dati FixedWidth assegna il valore configurato per la proprietà Valore predefinito dell'attributo del meta oggetto CxBlank al valore dell'attributo oggetto di business qualora incontri il valore di tale attributo meta oggetto CxBlank nel documento a larghezza fissa. E' necessario che gli oggetti di business abbiano almeno una chiave primaria che *non* contenga il valore CxBlank durante l'esecuzione.	valoreCxBlank
CxIgnore	Durante la conversione da un oggetto di business, il gestore dati FixedWidth scrive il valore configurato per la proprietà Valore predefinito dell'attributo meta oggetto CxIgnore al documento di larghezza fissa qualora incontri un attributo oggetto di business il cui valore sia CxIgnore. Durante la conversione di un oggetto di business, il gestore dati FixedWidth assegna il valore configurato per la proprietà Valore predefinito dell'attributo meta oggetto CxIgnore al valore dell'attributo oggetto di business qualora incontri il valore di questo attributo meta oggetto CxIgnore nel documento a larghezza fissa. E' necessario che gli oggetti di business abbiano almeno una chiave primaria che *non* contenga il valore CxIgnore durante l'esecuzione.	valoreCxIgnore
DummyKey	Attributo chiave richiesto dal sistema di integrazione business.	1
OmitObjectEventId	Valore booleano per determinare se includere o meno i dati ObjectEventId nelle conversioni oggetto di business in stringa e stringa in oggetto di business.	false

Tabella 56. Attributi meta oggetto secondario per il gestore dati FixedWidth (Continua)

Nome attributo meta oggetto	Significato	Valore predefinito fornito
PadCharacter	Indica gli spazi da aggiungere o rimuovere per l'allineamento. E' possibile specificare qualsiasi carattere come carattere pad.	#
Truncation	Imposta la rimozione dei caratteri. Se il valore è true, qualsiasi valore attributo nell'oggetto di business che è maggiore di MaxLength viene ridotto a MaxLength durante la richiesta di elaborazione. Se il valore è impostato su false, viene registrato un errore e la formattazione interrotta.	false
ObjectEventId	Placeholder non viene utilizzato dal gestore dati ma è richiesto dal sistema di integrazione business.	nessuno

Il "Valore predefinito fornito" nella colonna Tabella 56 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente nell'oggetto di business fornito. E' necessario verificare il proprio ambiente e impostare le proprietà del Valore predefinito di quegli attributi ai valori adatti al proprio sistema per i propri documenti FixedWidth. E' necessario assicurarsi che alla fine l'attributo ClassName abbia un valore predefinito.

Nota: Utilizzare Business Object Designer Express per modificare le definizioni dell'oggetto di business.

Requisiti degli oggetti di business

Il gestore dati FixedWidth crea presupposti riguardo alla struttura degli oggetti di business che gestisce. Perciò, durante la creazione di un oggetto di business utilizzando per la conversione il gestore dati FixedWidth, seguire queste regole:

- Assicurarsi che ogni attributo nella definizione dell'oggetto di business abbia un valore della proprietà MaxLength appropriato. Ciò assicura che il gestore dati FixedWidth possa elaborare correttamente la conversione dei dati da un oggetto di business ad un formato FixedWidth e viceversa.
- Assicurarsi che l'attributo ObjectEventId sia incluso in ogni oggetto di business a tutti i livelli di una gerarchia oggetto di business. Business Object Designer Express effettua questa operazione automaticamente quando salva una definizione dell'oggetto di business, ma sarebbe necessario confermare che il requisito è soddisfatto.

Struttura oggetto di business

Non ci sono requisiti circa la struttura degli oggetti di business per il gestore dati FixedWidth. Il gestore dati può elaborare qualsiasi oggetto di business finché MaxLength la proprietà dell'attributo abbia un valore.

Gli oggetti di business che il gestore dati elabora possono avere qualsiasi nome consentito dal sistema di integrazione business.

Proprietà dell'attributo oggetto di business

L'architettura dell'oggetto di business contiene varie proprietà che si applicano agli attributi. Tabella 57 descrive come il gestore dati FixedWidth interpreta tali proprietà e descrive come impostare le proprietà quando si modifica un oggetto di business.

Tabella 57. Proprietà dell'attributo per oggetti di business convertiti utilizzando il gestore dati FixedWidth

Nome proprietà	Descrizione
Nome	Ogni attributo dell'oggetto di business deve avere una denominazione univoca.
Tipo	Ogni attributo oggetto di business deve avere un tipo, come un Intero, una Stringa o il tipo di un oggetto di business secondario contenuto.
Chiave	Non utilizzato dal gestore dati FixedWidth.
MaxLength	Determina la larghezza del campo in cui il valore attributo è incluso.
Chiave esterna	Non utilizzato dal gestore dati FixedWidth.
Richiesto	Non utilizzato dal gestore dati FixedWidth.
Valore predefinito	Non utilizzato dal gestore dati FixedWidth.
Cardinalità	Supporta oggetti di cardinalità 1 e cardinalità n.

Informazioni sulle applicazioni per gli oggetti di business

Il gestore dati FixedWidth non richiede informazioni specifiche sull'applicazione negli oggetti di business o nei loro attributi. Il gestore dati, comunque, controlla l'esistenza della tag `cw_mo_`, che un oggetto di business potrebbe utilizzare per indicare qualsiasi meta oggetto secondario che il connettore utilizza. Il gestore dati ignora qualsiasi attributo identificato dalla tag `cw_mo_` nelle informazioni specifiche dell'applicazione dell'oggetto di business. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.

Utilizzo delle definizioni degli oggetti di business

Il gestore dati FixedWidth può convertire qualsiasi oggetto di business in una stringa FixedWidth così come un oggetto di business fornisce dati in una forma che ottempera ai requisiti del gestore dati. Il singolo requisito del gestore dati FixedWidth è che ciascun attributo oggetto di business ha un valore `MaxLength` specificato. Potrebbe essere necessario modificare oggetti di business esistenti per specificare un valore appropriato per `MaxLength`.

Sebbene gli oggetti di business esistenti che soddisfano tale requisito posso essere convertiti dal gestore dati FixedWidth, è buona norma creare i propri oggetti di business per ciascun tipo di dati da elaborare. Se si utilizza un oggetto di business di esempio o un oggetto di business sviluppato per supportare la stessa applicazione in un'altra implementazione, assicurarsi di modificare la definizione il necessario per includere solo gli attributi richiesti per l'implementazione per cui si sta sviluppando.

Perciò, per convertire oggetti di business esistenti in una forma che corrisponda dettagliatamente ai propri dati, modificare l'oggetto di business per fornire solo i dati richiesti dall'applicazione e le informazioni richieste dal gestore dati. Per adattare oggetti di business esistenti per l'utilizzo con il gestore dati FixedWidth, effettuare le seguenti operazioni:

1. Effettuare un'analisi funzionale dell'applicazione selezionata e paragonare i risultati con gli oggetti di business esistenti per determinare i campi richiesti di una definizione dell'oggetto di business.
2. Utilizzare Business Object Designer Express per aggiungere o eliminare, secondo necessità, attributi dalla definizione dell'oggetto di business.

Conversione oggetti di business in documenti FixedWidth

Per convertire un oggetto di business in un documento FixedWidth, il gestore dati FixedWidth esegue un loop tra gli attributi dell'oggetto di business in ordine sequenziale. Genera campi in una stringa a larghezza fissa ripetitivamente nell'ordine in cui gli attributi appaiono nell'oggetto di business e nei suoi elementi secondari.

Il gestore dati FixedWidth elabora oggetti di business in un documento FixedWidth nel modo seguente:

1. Il gestore dati crea una stringa a larghezza fissa per contenere i dati nell'oggetto di business.
2. Il gestore dati aggiunge il nome e il verbo dell'oggetto di business alla stringa a larghezza fissa. Il nome dell'oggetto di business può essere specificato come un argomento al metodo di conversione.
3. Il gestore dati verifica le informazioni specifiche dell'applicazione nella definizione dell'oggetto di business per determinare se ci sono meta oggetti secondari (quelli i cui nomi sono elencati nella tag `cw_mo_` delle informazioni specifiche dell'applicazione oggetto di business). Il gestore dati non include tali attributi nel documento FixedWidth FixedWidth. Per ulteriori informazioni riguardo alla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
4. Il gestore dati cerca l'attributo meta oggetto chiamato `OmitObjectEventId`. Se tale attributo è impostato su `true`, il gestore dati non include i dati `ObjectEventId` dell'oggetto di business nel documento `FixedWidth`.
5. Il gestore dati effettua un loop tra i rimanenti attributi dell'oggetto di business, aggiungendo la corretta variabile padding alla stringa per ciascun attributo semplice. Per gli attributi dell'array, il gestore dati effettua le seguenti operazioni:
   - Se l'attributo rappresenta un attributo a cardinalità singola, il gestore dati aggiunge il nome attributo e un conteggio di 1 alla stringa e quindi ripetitivamente elabora l'oggetto di business secondario per aggiungere i valori di ciascun attributo alla stringa.
   - Se l'attributo rappresenta un array a cardinalità multipla, il gestore dati aggiunge il nome attributo e l'oggetto conteggio secondario alla stringa e quindi ripetitivamente elabora ciascun oggetto di business secondario, aggiungendo i valori di ciascun attributo alla stringa.
6. Quando il gestore dati completa la conversione, restituisce i dati serializzati al richiedente. Il gestore dati restituisce i dati nel formato (`String` o `InputStream`) voluto dal richiedente.

Nota: Qualsiasi valore attributo nell'oggetto di business che ha una lunghezza maggiore di `MaxLength` è ridotto a `MaxLength` durante la richiesta di elaborazione se il valore della proprietà Valore predefinito della `l'attributo del meta oggetto Truncation` è impostato su `true`. Se `Truncation` è impostato su `false` e un valore attributo ha una lunghezza più grande di `MaxLength`, la formattazione termina e viene registrato un errore.

Conversione documenti FixedWidth in oggetti di business

Questa sezione fornisce le seguenti informazioni su come il gestore dati FixedWidth converte i documenti FixedWidth in oggetti di business:

- "Requisiti stringa FixedWidth"
- "Elaborazione dati serializzati"

Requisiti stringa FixedWidth

Durante la conversione di una stringa in un oggetto di business, il Il gestore dati FixedWidth crea i seguenti presupposti:

- Il nome oggetto di business appare come primo campo nei dati.
- Il verbo appare come secondo campo nei dati.
- Tutti gli attributi sono forniti nell'ordine in cui appaiono nella definizione dell'oggetto di business.
- L'attributo `ObjectEventId` è presente in ogni oggetto di business. Un'immissione per `ObjectEventId` è richiesta sempre per un oggetto di business, anche quando ha il valore `CxIgnore`, perché il gestore dati lo utilizza per distinguere tra istanze di un oggetto di business durante l'esecuzione.

Il formato per una stringa a larghezza fissa è il seguente:

```
[Bus_Obj_Name<padding_spazio_vuoto_dimensione>]
[Verb<padding_spazio_vuoto_dimensione>]
[Attr1<padding_spazio_vuoto_dimensione>]
[Attr2...<padding_spazio_vuoto_dimensione>]
[Number-of-child-object_instances<padding_spazio_vuoto_dimensione>]
[Child_Object_Name<padding_spazio_vuoto_dimensione>]
[Child_Object_Verb<padding_spazio_vuoto_dimensione>]
[Child_Object_Attr1<padding_spazio_vuoto_dimensione>]
[Child_Object_Attr2...<padding_spazio_vuoto_dimensione>]
<EndBO:Bus_Obj_Name>
```

In tale formato, i primi due campi (`Bus_Obj_Name` e `Verb`) sono allineati per creare campi di lunghezza specificata dalla `BONameSize` e attributi`BOVerbSize` nel meta oggetto secondario FixedWidth. I successivi attributi sono allineati per creare campi di lunghezza specificata nella la proprietà`MaxLength` per ciascun attributo oggetto di business.

Un campo utilizzato con il gestore dati FixedWidth deve essere lungo almeno otto caratteri se `CxIgnore` è un possibile valore per questo attributo. Se un attributo sicuramente non ha un valore `CxIgnore`, `MaxLength` può essere meno lungo di otto caratteri.

Quando un connettore legge un file in formato a grandezza fissa, la `CxIgnore` e `attributi meta oggetto CxBlank` devono essere configurati per generare i valori desiderati. Queste stringhe influiscono sugli attributi minimi `MaxLength`. Il valore minimo per `MaxLength` deve accontentare entrambi.

Elaborazione dati serializzati

Il gestore dati FixedWidth elabora un documento FixedWidth in un oggetto di business nel modo seguente:

1. Il gestore dati crea un oggetto di business per contenere i dati. Il tipo di oggetto di business viene trasferito o nel metodo di conversione dal connettore o il

gestore dati estrae il nome dell'oggetto di business dal primo campo della stringa. Se il tipo di parametro e il contenuto nei dati non coincidono, il gestore dati genera un errore.

2. Il gestore dati imposta il verbo nell'oggetto di business. Il gestore dati suppone che il verbo per l'oggetto di business di livello superiore sia nel secondo campo dei dati.
3. Il gestore dati determina se ci sono meta oggetti secondari (quelli i cui nomi sono elencati nella tag `cw_mo_` delle informazioni specifiche dell'applicazione dell'oggetto di business). Il gestore dati *non* effettua l'elaborazione per compilare questi attributi dell'oggetto di business. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
4. Il gestore dati ricerca l'attributo meta oggetto chiamato `OmitObjectEventId`. Se tale attributo è impostato su `true`, il gestore dati non effettua l'elaborazione per compilare l'attributo `ObjectEventId`.
5. Per impostare i rimanti attributi oggetto di business, il gestore dati analizza i dati basati su `MaxLength` di ciascun attributo come specificato nella definizione dell'oggetto di business. Estrae valori attributo dai dati e popola i valori degli attributi semplici nell'oggetto di business.

   Il gestore dati elabora gli attributi dell'array nel seguente modo:

   - Se l'attributo è un attributo a cardinalità singola, il gestore dati analizza ripetitivamente l'elenco attributi, imposta i valori attributo e aggiunge l'oggetto di business secondario all'oggetto di business di livello superiore.
   - Se l'attributo è un array a cardinalità multipla, il gestore dati analizza ripetitivamente gli attributi in ogni elenco attributi secondari e aggiunge l'oggetto di business secondario all'oggetto di business di livello superiore.

Capitolo 7. Gestore dati delimitato

Il gestore dati delimitato converte oggetti di business in stringhe e flussi di formato delimitato e viceversa. Questo capitolo descrive come il gestore dati delimitato elabora dati delimitati e come definire gli oggetti di business che devono essere elaborati dal gestore dati. E' possibile utilizzare queste informazioni per modificare oggetti di business esistenti o implementare nuovi oggetti di business che si adattino ai requisiti del gestore dati. Inoltre, questo capitolo descrive come configurare il gestore dati delimitato. Questo capitolo contiene le sezioni seguenti:

Nota: Il gestore dati delimitato è uno dei gestori dati base contenuti nel file `CwDataHandler.jar`. Per ulteriori informazioni su come installare questo gestore dati e su dove memorizzare il suo codice sorgente, fare riferimento a Capitolo 2, "Installazione e configurazione del Gestore Dati", a pagina 21.

Panoramica

Il gestore dati delimitato è un modulo di conversione dati il cui ruolo primario è di convertire gli oggetti di business in stringhe o flussi di formato delimitato e viceversa. Una stringa o un flusso di formato delimitato è un dato serializzato con `testo/delimitato` tipo MIME. Il gestore dati analizza i dati di testo in base ad uno specifico delimitatore che separa i campi individuali dei dati di un oggetto di business. Tale tipo di conversione dati è utilizzato principalmente dove l'efficienza di lettura della macchina è più importante.

Il meta oggetto del connettore di livello superiore predefinito (`MO_DataHandler_Default`) supporta il tipo MIME `text/delimited`. Perciò, un connettore configurato per l'utilizzo del meta oggetto `MO_DataHandler_Default` può richiamare il gestore dati delimitato. Se InterChange Server Express viene utilizzato ed è necessario che un client di accesso sia in grado di richiamare tale gestore dati, è necessario modificare il meta oggetto del server di livello superiore (`MO_Server_DataHandler`) per supportare il tipo MIME `text/delimited`. Per ulteriori informazioni, consultare "Modifica del meta-oggeto di livello superiore" a pagina 197.

Caratteristiche del gestore dati delimitato

Il gestore dati delimitato consente di impostare le seguenti stringhe:

- Il delimitatore—Il gestore dati utilizza un delimitatore per separare i diversi campi nei dati delimitati. E' possibile impostare l'attributo del meta oggetto `delimitatore` sul delimitatore desiderato per i propri dati. Per impostazione predefinita, il gestore dati utilizza un carattere tilde (~) come delimitatore che è la dimensione della proprietà dell'attributo `MaxLength` per determinare come leggere e scrivere i dati. `MaxLength` è una proprietà dell'attributo dell'oggetto di business che definisce il numero massimo di caratteri del valore dell'attributo, inclusa la variabile padding che consente il testo giustificato a destra o a sinistra. `MaxLength` è letto dalla definizione di oggetto di business nel repository. Di

conseguenza, il requisito principale per un oggetto di business è che MaxLength sia impostato appropriatamente per ogni attributo della stringa.

- La stringa escape—Il gestore dati utilizza la stringa escape per configurare una stringa per evitare il delimitatore e le stringhe escape. La stringa escape consente ai dati del valore dell'attributo di contenere stringhe tipo delimitatore e tipo escape. E' possibile impostare l'attributo del meta oggetto Escapeper configurare la stringa escape. Per impostazione predefinita, il gestore dati utilizza il carattere barra rovesciata (\) come stringa escape.

Oggetto di business ed elaborazione stringa

Il gestore dati delimitato effettua le operazioni elencate in Tabella 58.

Tabella 58. Operazioni dati per il gestore dati delimitato

operazione gestore dati	Per ulteriori informazioni
Riceve un oggetto di business dal richiedente, converte l'oggetto di business in una stringa o flusso delimitato e passa i dati serializzati al richiedente.	"Conversione degli oggetti di business in dati delimitati" a pagina 152
Riceve una stringa o un flusso dal richiedente, crea un oggetto di business e invia di nuovo l'oggetto di business al richiedente. Riceve una stringa o un flusso delimitato del richiedente, crea un oggetto di business e lo passa al richiedente.	"Conversione dei dati delimitati in oggetti di business" a pagina 152

Configurazione del gestore dati delimitato

Per configurare il gestore dati delimitato, procedere nel modo seguente:

- Immettere i valori appropriati per gli attributi del meta oggetto secondario delimitato.
- Creare o modificare le definizioni oggetto di business in modo che supportino il gestore dati.

Ognuno di questi passaggi è descritto con maggiori dettagli nelle sezioni seguenti.

Configurazione del meta oggetto secondario delimitato

Per configurare un gestore dati delimitato, è necessario assicurarsi che le sue informazioni di configurazione siano fornite nel meta oggetto secondario delimitato. Per il gestore dati delimitato, IBM fornisce il meta oggetto secondario MO_DataHandler_DefaultDelimitedConfig. Ogni attributo in questo meta oggetto definisce una proprietà di configurazione per il gestore dati delimitato. Tabella 59 descrive gli attributi in questo meta oggetto secondario.

Tabella 59. Attributi meta oggetto secondario per il gestore dati delimitato

Nome attributo meta oggetto	Significato	Valore predefinito fornito
ClassName	Nome della classe del gestore dati da caricare per l'utilizzo con il tipo MIME specificato. Il meta oggetto del gestore dati di livello superiore ha un attributo il cui nome corrisponde la tipo MIME specificato e il cui tipo è il meta oggetto secondario delimitato (descritto da Tabella 59).	com.crossworlds. DataHandlers. text.delimited
CxBlank	Stabilisce il valore equivalente nei dati delimitati per il valore dell'attributo dell'oggetto di business speciale, Blank (la costante CxBlank). Per ulteriori informazioni, consultare "CxBlank" a pagina 150.	costante CxBlank (spazio vuoto)

Tabella 59. Attributi meta oggetto secondario per il gestore dati delimitato (Continua)

Nome attributo meta oggetto	Significato	Valore predefinito fornito
CxIgnore	Stabilisce il valore equivalente nei dati delimitati per il valore dell'attributo dell'oggetto di business speciale, Ignore (la costante CxIgnore). Per ulteriori informazioni, consultare "CxIgnore" a pagina 150.	costante CxIgnore (stringa vuota)
delimitatore	Stringa utilizzata per separare i valori negli attributi dell'oggetto di business durante la scrittura di dati dell'oggetto di business nei file, cioè di separare i campi di dati corrispondenti agli attributi durante la conversione di un file in un oggetto di business. Tale valore può contenere più caratteri.	~ (tilde)
DummyKey	Attributo chiave richiesto dal sistema di integrazione business.	1
Escape	Stringa utilizzata per evitare il delimitatore e i caratteri escape se si trovano in un valore dell'attributo dell'oggetto di business. Tale valore può essere lungo solo un carattere.	\ (barra inversa)
OmitObjectEventId	Valore booleano per determinare se includere o meno i dati ObjectEventId nelle conversioni oggetti di business in stringa e stringa in oggetto di business.	falso
ObjectEventId	Attributo Placeholder richiesto dal sistema di integrazione business.	nessuno

Il "Valore predefinito fornito" colonna in Tabella 59 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente nell'oggetto di business fornito. E' necessario verificare il proprio ambiente e impostare le proprietà del Valore predefinito di quegli attributi sui valori adatti al proprio sistema per i propri documenti delimitati. E' necessario assicurarsi che alla fine l'attributo ClassName abbia un valore predefinito.

Nota: Utilizzare Business Object Designer Express per modificare le definizioni di un oggetto di business.

Requisiti dell'oggetto di business

Il gestore dati delimitato crea presupposti riguardo gli oggetti di business che gestisce. Perciò, durante la creazione di un oggetto di business utilizzando per la conversione il gestore dati delimitato, segue queste regole:

- Assicurarsi che ogni attributo dell'oggetto di business ha un nome per garantire che il gestore dati delimitato possa convertire in maniera appropriata i dati.
- Assicurarsi che l'attributo ObjectEventId sia incluso in ogni oggetto di business a tutti i livelli di una gerarchia oggetto di business. Business Object Designer Express effettua questa operazione automaticamente quando salva una definizione oggetto di business, ma sarebbe necessario confermare che il requisito è soddisfatto.

L'attributo delimitatore nel meta oggetto secondario configura il delimitatore utilizzato per separare i campi dell'attributo. Il valore predefinito è il carattere tilde (~).

E' possibile impostare l'attributo del meta oggetto secondario Escape per configurare una stringa per evitare il delimitatore e le stringhe escape. La stringa escape consente ai dati del valore dell'attributo di contenere stringhe tipo delimitatore e tipo escape.

Struttura oggetto di business

Non ci sono requisiti circa la struttura degli oggetti di business per il gestore dati delimitato. Il gestore dati può elaborare ogni oggetto di business.

Gli oggetti di business che il gestore dati elabora possono avere qualsiasi nome consentito da InterChange Server Express.

Proprietà dell'attributo oggetto di business

L'architettura dell'oggetto di business contiene varie proprietà che si applicano agli attributi. Tabella 60 descrive come il gestore dati delimitato interpreta numerose di queste proprietà e descrive come impostare le proprietà quando si modifica un oggetto di business.

Tabella 60. Proprietà attributo per oggetti di business convertiti utilizzando il gestore dati delimitato

Nome proprietà	Descrizione
Name	E' necessario che ogni attributo dell'oggetto di business abbia un nome univoco.
Type	Ogni attributo oggetto di business deve avere un tipo, come un Intero, una Stringa o il tipo di un oggetto di business secondario contenuto.
Key	Non utilizzato dal gestore dati delimitato.
MaxLength	Non utilizzato dal gestore dati delimitato.
Foreign Key	Non utilizzato dal gestore dati delimitato.
Required	Non utilizzato dal gestore dati delimitato.
Valore predefinito	Non utilizzato dal gestore dati delimitato.
Cardinalità	Supporta oggetti di cardinalità 1 e cardinalità n.

E' possibile che gli attributi negli oggetti di business abbiano valori speciali di `CxIgnore` o `CxBlank`. Il gestore dati delimitato effettua speciali passaggi di elaborazione quando gli attributi hanno tali valori, come descritto nelle seguenti sezioni.

CxIgnore: L'attributo del meta oggetto `CxIgnore` stabilisce il valore equivalente nei dati delimitati data per il valore dell'attributo `Ignore` (la costante `CxIgnore`). Per impostazione predefinita, l'attributo del meta oggetto `CxIgnore` è impostato sul valore della costante `CxIgnore`. Il gestore dati utilizza l'attributo del meta oggetto `CxIgnore` come segue:

- Durante la conversione *da* un oggetto di business, il gestore dati delimitato scrive il valore dell'attributo del meta oggetto `CxIgnore` (il valore configurato per la relativa proprietà Valore predefinito) nei dati delimitati qualora incontri un attributo dell'oggetto di business con la costante `CxIgnore` come relativo valore dell'attributo.
- Durante la conversione *in* un oggetto di business, il gestore dati delimitato assegna la costante `CxIgnore` al valore dell'attributo dell'oggetto di business qualora incontri il valore dell'attributo del meta oggetto `CxIgnore` (il valore configurato per la relativa proprietà Valore predefinito) nei dati delimitati.

Nota: E' necessario che gli oggetti di business abbiano almeno una chiave primaria che *non* contenga il valore `CxIgnore` durante l'esecuzione.

CxBlank: L'attributo del meta oggetto `CxBlank` stabilisce il valore equivalente nei dati delimitati data per il valore dell'attributo `Blank` (la costante `CxBlank`). Per

impostazione predefinita, l'attributo del meta oggetto CxBlank è impostato sul valore della costante CxBlank. Il gestore dati utilizza l'attributo del meta oggetto CxBlank come segue:

- Durante la conversione *da* un oggetto di business, il gestore dati delimitato scrive il valore dell'attributo del meta oggetto CxBlank (il valore configurato per la proprietà Valore predefinito) nei dati delimitati qualora incontri un attributo dell'oggetto di business con la costante CxBlankcome relativo valore dell'attributo.
- Durante la conversione *in* un oggetto di business, il gestore dati delimitato assegna la costante CxBlank al valore dell'attributo dell'oggetto di business qualora incontri il valore dell'attributo del meta oggetto CxBlank (il valore configurato per la relativa proprietà Valore predefinito) nei dati delimitati.

Nota: E' necessario che gli oggetti di business abbiano almeno una chiave primaria che *non* contenga il valore CxBlank durante l'esecuzione.

Informazioni sulle applicazioni per gli oggetti di business

Il gestore dati delimitato non richiede informazioni specifiche sull'applicazione negli oggetti di business o nei loro attributi. Il gestore dati, comunque, controlla l'esistenza della tag cw_mo_, che un oggetto di business potrebbe utilizzare per indicare qualsiasi meta oggetto secondario che il connettore utilizza. Il gestore dati ignora qualsiasi attributo identificato dalla tag cw_mo_ nelle informazioni specifiche dell'applicazione dell'oggetto di business. Per ulteriori informazioni sulla tag cw_mo_, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.

Utilizzo delle definizioni degli oggetti di business

Il gestore dati delimitato può convertire qualsiasi oggetto di business in una stringa delimitata sempre che un oggetto di business fornisca dati in una forma che ottemperi ai requisiti del gestore dati. L'unico requisito del gestore dati delimitato è che se il gestore dati deve effettuare la lettura in un file delimitato, ogni campo individuale sia separato dal delimitatore configurato.

Sebbene gli oggetti di business esistenti che soddisfano tale requisito posso essere convertiti dal gestore dati delimitato, è buona norma creare i propri oggetti di business per ciascun tipo di dati da elaborare. Se si utilizza un oggetto di business di esempio o un oggetto di business sviluppato per supportare la stessa applicazione in un'altra implementazione, assicurarsi di modificare la definizione il necessario per includere solo gli attributi richiesti per l'implementazione per cui si sta sviluppando.

Perciò, per convertire oggetti di business esistenti in una forma che corrisponda dettagliatamente ai propri dati, modificare l'oggetto di business per fornire solo i dati richiesti dall'applicazione e le informazioni richieste dal gestore dati. Per adattare oggetti di business esistenti per l'utilizzo con il gestore dati delimitato, effettuare le seguenti operazioni:

1. Effettuare un'analisi funzionale dell'applicazione selezionata e paragonare i risultati con gli oggetti di business esistenti per determinare i campi richiesti di una definizione oggetto di business.
2. Utilizzare Business Object Designer Express per aggiungere o eliminare, secondo necessità, attributi dalla definizione dell'oggetto di business.

Conversione degli oggetti di business in dati delimitati

Per effettuare la conversione di un oggetto di business in una stringa, il gestore dati delimitato effettua un loop tra gli attributi dell'oggetto di business in ordine sequenziale. Genera formattazioni delimitate in maniera ricorrente nell'ordine in cui gli attributi si trovano nell'oggetto di business e nei relativi oggetti secondari. Il nome dell'oggetto di business viene passato come un argomento al metodo di conversione.

Il gestore dati delimitato effettua l'elaborazione degli oggetti di business nei dati delimitati come segue:

1. Il gestore dati crea una stringa per contenere i dati nell'oggetto di business.
2. Il gestore dati aggiunge il nome dell'oggetto di business come primo token nella stringa e aggiunge il verbo come secondo token nella stringa.
3. Il gestore dati esamina le informazioni dell'applicazione nella definizione dell'oggetto di business per determinare se vi siano meta oggetti secondari (i cui nomi sono elencati nella tag `cw_mo_` delle informazioni dell'applicazione dell'oggetto di business). Il gestore dati *non* include tali attributi nei dati delimitati. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
4. Il gestore dati ricerca l'attributo del meta oggetto chiamato `OmitObjectEventId`. Se tale attributo è impostato su `true`, il gestore dati non include i dati dell'oggetto di business `ObjectEventId` nei dati delimitati.
5. Il gestore dati effettua un loop tra i rimanenti attributi dell'oggetto di business aggiungendo valori ad ogni attributo semplice per la stringa e aggiungendo il delimitatore configurato dopo ogni attributo. Per gli attributi del contenitore, il gestore dati effettua le seguenti operazioni:
   - Se l'attributo è un contenitore a cardinalità 1, il gestore dati aggiunge il conteggio dell'attributo alla stringa e quindi elabora in maniera ricorrente 'oggetto di business secondario per aggiungere valori per ciascun attributo.
   - Se l'attributo è un contenitore a cardinalità n, il gestore dati aggiunge il conteggio degli oggetti secondari nel contenitore alla stringa e quindi elabora in maniera ricorrente ogni oggetto di business secondario, aggiungendo valori per ciascun attributo nella stringa.
6. Quando il gestore dati completa la conversione, restituisce i dati serializzati al richiedente. Il gestore dati restituisce i dati nel formato (stringa o InputStream) desiderato dal richiedente.

Il formato che il gestore dati genera corrisponde al seguente modello:

```
Bus_Obj_Name<delimiter>Verb<delimiter>Attr1<delimiter>Attr2<delimiter>
Number_of_child_object_instances<delimiter>Child_Object_Name<delimiter>Verb
<delimiter>Attr1<delimiter>Attr2<EndBO:Bus_Obj_Name>
```

Una stringa escape viene apposta davanti ad ogni stringa del tipo delimitatore in un valore dell'attributo. La stringa escape viene configurata utilizzando l'attributo `Escape` del meta oggetto secondario.

Conversione dei dati delimitati in oggetti di business

Questa sezione fornisce le seguenti informazioni su come il gestore dati delimitati effettua la conversione dei dati delimitati in oggetti di business:

- "Requisiti della stringa delimitata" a pagina 153
- "Elaborazione dati serializzati" a pagina 153

Requisiti della stringa delimitata

Quando effettua la conversione di una stringa o flusso, il gestore dati delimitato crea i seguenti presupposti:

- I dati contengono il delimitatore specificato nell'attributo del meta oggetto `delimitatore`.
- Il nome dell'oggetto di business viene visualizzato nel primo campo nei dati.
- Il verbo appare come secondo campo nei dati.
- Gli attributi si trovano nell'ordine in cui vengono visualizzati nella definizione di oggetto di business.
- Tutti gli oggetti in un contenitore secondario sono dello stesso tipo.
- I dati contengono un token che rappresenta il numero di oggetti secondari in ogni contenitore a cardinalità `n`.
- L'attributo `ObjectEventId` è presente in ogni oggetto di business. Un'immissione per `ObjectEventId` è richiesta sempre per un oggetto di business, anche quando ha il valore `CxIgnore`, perché il gestore dati lo utilizza per distinguere tra istanze di un oggetto di business durante l'esecuzione.

Se è presente più di un oggetto di business nei dati, assicurarsi di non introdurre nessun nuovo carattere tra questi (ad esempio uno spazio, un carattere tab, una nuova riga o un ritorno a capo).

Quando il gestore dati delimitato legge un file in formato delimitato, effettua i seguenti passaggi speciali di elaborazione per assegnare ad un attributo dell'oggetto di business il valore dell'attributo `CxIgnore` o `CxBlank`:

- Assegna la costante `CxIgnore` (`nulla`) come il valore dell'attributo corrispondente qualora incontri una qualsiasi delle seguenti condizioni nei dati delimitati:
  - Il valore dell'attributo del meta oggetto `CxIgnore` (valore configurato per la relativa proprietà Valore predefinito)
  - Il valore di una stringa vuota (″ ″)
- Assegna la costante `CxBlank` come il valore dell'attributo solo quando l'attributo del meta oggetto `CxBlank` è configurato ed incontra tale valore configurato nei corrispondenti dati delimitati.

Nota: Assicurarsi che la stringa escape e il delimitatore abbiano valori differenti come configurato dagli attributi del meta oggetto `Escape` e `delimitatore` nel meta oggetto secondario del gestore dati delimitatore.

La seguente riga mostra l'esempio di una stringa in formato delimitato. La sintassi è:

```
Bus_Obj_Name<delimiter>Verb<delimiter>Attr1<delimiter>Attr2<delimiter>
Number_of_child_object_instances<delimiter>Child_Object_Name<delimiter>
Verb<delimiter>Attr1<delimiter>Attr2<EndBO:Bus_Obj_Name>
```

Il seguente esempio utilizza un delimitatore tilde (~):

```
Customer~Create~p1~p2~p3~1~CustomerAddress~Create~q1~q2~q3~q4~q5~q6~q7~q8~q9~q10~3~
PhoneInfo~Create~r1~r2~r3~r4~r5~r6~r7~PhoneInfo~Create~r1~r2~r3~r4~r5~r6~r7~
PhoneInfo~Create~r1~r2~r3~r4~r5~r6~r7
```

Elaborazione dati serializzati

Il gestore dati delimitato effettua l'elaborazione dei dati delimitati in un oggetto di business come segue:

1. Il gestore dati prende il nome dell'oggetto di business dal primo token nei dati e crea un oggetto di business per contenere i dati.
2. Il gestore dati imposta il verbo nell'oggetto di business. Il gestore dati assume che il verbo per l'oggetto di business di livello superiore sia nel secondo token nei dati delimitati. Notare che gli oggetti di business secondari possono non avere verbi impostati.
3. Il gestore dati determina se ci sono meta oggetti secondari (quelli i cui nomi sono elencati nella tag `cw_mo_` delle informazioni specifiche dell'applicazione dell'oggetto di business). Il gestore dati *non* effettua l'elaborazione per popolare questi attributi dell'oggetto di business. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
4. Il gestore dati ricerca l'attributo del meta oggetto chiamato `OmitObjectEventId`. Se tale attributo è impostato su `vero`, il gestore dati non effettua l'elaborazione per popolare l'attributo `ObjectEventId`.
5. Il gestore dati analizza i dati e immette i valori degli attributi semplici rimanenti nell'oggetto di business con i valori del token dei dati. Il gestore dati elabora gli attributi del contenitore come segue:
   - Se l'attributo è a cardinalità singola, il gestore dati analizza in maniera ricorrente i token dell'attributo nella stringa, imposta i valori dell'attributo nell'oggetto di business e aggiunge il contenitore dell'oggetto di business secondario dall'oggetto di business principale.
   - Se l'attributo è a cardinalità multipla, il gestore dati analizza in maniera ricorrente i token dell'attributo per ogni oggetto secondario, imposta i valori dell'attributo nell'oggetto di business secondario e aggiunge il contenitore dell'oggetto di business secondario all'oggetto di business principale.

Capitolo 8. Gestore dati NameValue

Il gestore dati NameValue converte un oggetto business in uno specifico formato dati serializzati e converte i dati serializzati in uno specifico formato in un oggetto di business. Questo capitolo descrive come il gestore dati NameValue elabora dati NameValue e come definire gli oggetti di business che devono essere elaborati dal gestore dati. E' possibile utilizzare queste informazioni per implementare gli oggetti di business che si adattano ai requisiti del gestore dati NameValue. Questo capitolo descrive, inoltre, come configurare il gestore dati NameValue. Questo capitolo contiene le sezioni seguenti:

Nota: Il gestore dati NameValue è uno dei gestori dati base contenuti nel file `CwDataHandler.jar`. Per ulteriori informazioni su come installare questo gestore dati e su dove memorizzare il suo codice sorgente, fare riferimento a Capitolo 2, "Installazione e configurazione del Gestore Dati", a pagina 21.

Panoramica

Il gestore dati NameValue è un modulo di conversione dati il cui ruolo primario è di convertire gli oggetti di business in stringhe o flussi formattati in coppie nome-valore. Una stringa o un flusso formattato per NameValue è un dato serializzato con `text/namevalue` tipo MIME. Il gestore dati NameValue può essere utilizzato per generare un file per l'oggetto di business a scopo di verifica.

Il gestore dati analizza i dati serializzati basati su determinati campi. Ad esempio, un file di testo per tale gestore dati contiene i campi che identificano il tipo di oggetto di business (BusinessObject=BOname), il verbo (Verb=verbName), il numero di attributi (AttributeCount=numericValue) e i valori dell'attributo (`AttributeName=`*`Value`*). Il gestore dati utilizza le informazioni del valore del nome per analizzare i dati.

Il meta oggetto del connettore di livello superiore predefinito (`MO_DataHandler_Default`) supporta il tipo MIME `text/namevalue`. Perciò, un connettore configurato per l'utilizzo del meta oggetto `MO_DataHandler_Default` può richiamare il gestore dati NameValue. Perché un client di accesso sia in grado di richiamare tale gestore dati durante l'utilizzo di InterChange Server Express, è necessario modificare il meta oggetto `MO_Server_DataHandler` affinché supporti il tipo MIME `text/namevalue`. Per ulteriori informazioni, consultare "Modifica del meta-oggeto di livello superiore" a pagina 197.

. Il gestore dati NameValue effettua le operazioni elencate in Tabella 61.

Tabella 61. Operazioni dati per il gestore dati NameValue

operazione gestore dati	Per ulteriori informazioni
Riceve un oggetto di business dal richiedente, converte l'oggetto di business in una stringa o flusso NameValue e passa i dati serializzati al richiedente.	"Conversione degli oggetti di business in dati NameValue" a pagina 160

Tabella 61. Operazioni dati per il gestore dati NameValue (Continua)

operazione gestore dati	Per ulteriori informazioni
Riceve una stringa o un flusso dal richiedente, crea un oggetto di business e invia di nuovo l'oggetto di business al richiedente.	"Conversione dei dati NameValue in oggetti di business" a pagina 161

Configurazione del gestore dati NameValue

Per configurare il gestore dati NameValue, procedere nel modo seguente:

- Immettere i valori appropriati per gli attributi del meta oggetto secondario NameValue.
- Creare o modificare le definizioni oggetto di business in modo che supportino il gestore dati.

Ognuno di questi passaggi è descritto con maggiori dettagli nelle sezioni seguenti.

Configurazione del meta oggetto secondario NameValue

Per configurare un gestore dati NameValue, è necessario assicurarsi che le sue informazioni di configurazione siano fornite nel meta oggetto secondario NameValue. Per il gestore dati NameValue, IBM fornisce il `MO_DataHandler_DefaultNameValueConfig` meta oggetto secondario. Ogni attributo in tale meta oggetto definisce una proprietà di configurazione per il gestore dati NameValue. Tabella 62 descrive gli attributi per questo meta oggetto secondario.

Tabella 62. Attributi del meta oggetto secondario per il gestore dati NameValue

Nome attributo meta oggetto	Descrizione	Valore predefinito fornito
`ClassName`	Nome della classe del gestore dati da caricare per l'utilizzo con il tipo MIME specificato. Il meta oggetto del gestore dati di livello superiore ha un attributo il cui nome corrisponde al tipo MIME specificato e il cui tipo è il meta oggetto secondario NameValue (descritto dal Tabella 62).	`com.crossworlds.DataHandlers.text.namevalue`
`CxBlank`	Stabilisce il valore equivalente nei dati NameValue per il valore dell'attributo dell'oggetto di business speciale, Blank (la costante `CxBlank`). Per ulteriori informazioni, consultare "CxBlank" a pagina 159.	costante `CxBlank`
`CxBlankValue`	*Questo attributo non è più utilizzato.* Utilizzare l'attributo del meta oggetto `CxBlank` (vedere sopra) per indicare al gestore dati come i dati NameValue rappresentano il valore dell'attributo `CxBlank`.	spazio vuoto
`CxIgnore`	Stabilisce il valore equivalente nei dati NameValue per il valore dell'attributo dell'oggetto di business speciale, Ignore (la costante `CxIgnore`). Per ulteriori informazioni, consultare "CxIgnore" a pagina 158.	costante `CxIgnore`
`DefaultVerb`	verbo dell'oggetto di business	`Creare`
`DummyKey`	Attributo chiave richiesto dal sistema di integrazione business.	`1`
`SkipCxIgnore`	Durante l'elaborazione delle richieste, l'elaborazione del valore dell'attribute speciale `CxIgnore` è basato sull'attributo del meta oggetto `SkipCxIgnore`. Per ulteriori informazioni, consultare "CxIgnore" a pagina 158.	`falso`
`ValidateAttrCount`	Determina se il gestore dati ricerca (o aggiunge alla stringa di output) un token contenente un numero di attributi nei dati dell'oggetto di business.	`vero`

Tabella 62. Attributi del meta oggetto secondario per il gestore dati NameValue (Continua)

Nome attributo meta oggetto	Descrizione	Valore predefinito fornito
`ObjectEventId`	Placeholder non viene utilizzato dal gestore dati ma è richiesto dal sistema di integrazione business.	nessuno

Il "Valore predefinito fornito" colonna in Tabella 62 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente nell'oggetto di business fornito. E' necessario esaminare il proprio ambiente ed impostare le proprietà del Valore predefinito di tali attributi sui valori appropriati per il proprio sistema e per i documenti in formato coppia nome-valore.

Nota: Utilizzare Business Object Designer Express per modificare le definizioni di un oggetto di business.

Requisiti dell'oggetto di business

Il gestore dati NameValue crea presupposti riguardo gli oggetti di business che gestisce. Perciò, quando si passa un oggetto di business per la conversione con il gestore dati NameValue, seguire queste regole:

- Assicurarsi che ogni attributo dell'oggetto di business abbia una proprietà `Name`. Ciò assicura che il gestore dati NameValue possa elaborare correttamente la conversione dei dati da un oggetto di business ad un formato NameValue e viceversa.
- Assicurarsi che l'attributo `ObjectEventId` sia incluso in ogni oggetto di business a tutti i livelli di una gerarchia oggetto di business. Business Object Designer Express effettua questa operazione automaticamente quando salva una definizione oggetto di business, ma sarebbe necessario confermare che il requisito è soddisfatto.

Struttura oggetto di business

Non ci sono requisiti circa la struttura degli oggetti di business per il gestore dati NameValue. Il gestore dati può elaborare ogni oggetto di business.

Gli oggetti di business che il gestore dati elabora possono avere qualsiasi nome consentito dal sistema di integrazione business.

Proprietà dell'attributo oggetto di business

L'architettura dell'oggetto di business contiene varie proprietà che si applicano agli attributi. Tabella 63 descrive come il gestore dati NameValue interpreta numerose di queste proprietà e descrive come impostare le proprietà durante la modifica della definizione di un oggetto di business.

Tabella 63. Proprietà dell'attributo per oggetti di business convertiti utilizzando il gestore dati NameValue

Nome proprietà	Descrizione
Name	Ogni attributo dell'oggetto di business deve avere una denominazione univoca.
Type	E' necessario che ogni attributo dell'oggetto di business abbia un tipo, come ad esempio un numero intero, una stringa o il tipo di un oggetto di business secondario contenuto. E' necessario che tutti gli attributi semplici siano del tipo stringa.
Key	Non utilizzato dal gestore dati NameValue.
MaxLength	Non utilizzato dal gestore dati NameValue.
Foreign Key	Non utilizzato dal gestore dati NameValue.
Required	Non utilizzato dal gestore dati NameValue.
Valore predefinito	Non utilizzato dal gestore dati NameValue.
Cardinalità	Supporta oggetti di cardinalità 1 e cardinalità n.

E' possibile che gli attributi negli oggetti di business abbiano valori speciali di `CxIgnore` o `CxBlank`. Il gestore dati NameValue effettua speciali passaggi di elaborazione quando gli attributi hanno tali valori, come descritto nelle seguenti sezioni.

CxIgnore: L'attributo del meta oggetto `CxIgnore` stabilisce il valore equivalente nei dati NameValue data per il valore dell'attributo `Ignore` (la costante `CxIgnore`). Per impostazione predefinita, l'attributo del meta oggetto `CxIgnore` è impostato sul valore della costante `CxIgnore`. Il gestore dati utilizza l'attributo del meta oggetto `CxIgnore` come segue:

- Durante la conversione *da* un oggetto di business, il gestore dati NameValue scrive il valore dell'attributo del meta oggetto `CxIgnore` (il valore configurato per la relativa proprietà Valore predefinito) nei dati NameValue qualora incontri un attributo dell'oggetto di business con la costante `CxIgnore` come relativo valore dell'attributo.
- Durante la conversione *in* un oggetto di business, il gestore dati NameValue assegna la costante `CxIgnore` al valore dell'attributo dell'oggetto di business qualora incontri una qualsiasi delle seguenti condizioni dati NameValue:
  - Il valore dell'attributo del meta oggetto `CxIgnore` (valore configurato per la relativa proprietà Valore predefinito)
  - Il valore di una stringa vuota
  - Nessun valore corrispondente

Nota: E' necessario che gli oggetti di business abbiano almeno una chiave primaria che *non* contenga il valore `CxIgnore` durante l'esecuzione.

E' possibile configurare come il gestore dati NameValue elabori gli attributi con un valore dell'attributo di `CxIgnore`. Ad esempio:

- E' possibile configurare se si desidera che il gestore dati elabori o ignori gli attributi con un valore `CxIgnore` durante l'elaborazione delle richieste.
- E' possibile decidere di non creare una voce per gli attributi con un valore `CxIgnore` in modo che il connettore possa elaborare i dati dell'oggetto di business correttamente durante la notifica degli eventi. Questo è utile durante la creazione di un file fittizio per un tipo dell'oggetto.

Durante l'elaborazione delle richieste, il gestore dati crea una versione serializzata dell'oggetto di business. A questo punto, l'elaborazione del valore dell'attributo speciale CxIgnore è basato sull'attributo del meta oggetto secondario SkipCxIgnore, come segue:

- Quando SkipCxIgnore è impostato su false, il gestore dati scrive il valore di questo attributo del meta oggetto CxIgnore nei dati NameValue qualora incontri un attributo dell'oggetto di business con la costante CxIgnore come il valore dell'attributo.
- Se SkipCxIgnore è impostato su true, il gestore dati ignora tutti gli attributi con un valore di CxIgnore e *non* genera per loro nessun dato NameValue.

Nota: Con SkipCxIgnore impostato su true, il gestore dati NameValue *non* è bidirezionale; ossia, non può effettuare conversioni stringa in oggetto di business su stringhe generate durante le conversioni oggetto di business in stringa.

CxBlank: L'attributo del meta oggetto CxBlank stabilisce il valore equivalente nei dati NameValue data per il valore dell'attributo Blank (la costante CxBlank). Per impostazione predefinita, l'attributo del meta oggetto CxBlank è impostato sul valore della costante CxBlank. Il gestore dati utilizza l'attributo del meta oggetto CxBlank come segue:

- Durante la conversione *da* un oggetto di business, il gestore dati NameValue scrive il valore dell'attributo del meta oggetto CxBlank (il valore configurato per la proprietà Valore predefinito) nei dati NameValue qualora incontri un attributo dell'oggetto di business con la costante CxBlankcome relativo valore dell'attributo.
- Durante la conversione *in* un oggetto di business, il gestore dati NameValue assegna la costante CxBlank al valore dell'attributo dell'oggetto di business qualora incontri il valore dell'attributo del meta oggetto CxBlank (il valore configurato per la relativa proprietà Valore predefinito) nei dati NameValue.

Nota: E' necessario che gli oggetti di business abbiano almeno una chiave primaria che *non* contenga il valore CxBlank durante l'esecuzione.

Informazioni sulle applicazioni per gli oggetti di business

Il gestore dati NameValue non richiede informazioni specifiche sull'applicazione negli oggetti di business o nei loro attributi. Il gestore dati, comunque, controlla l'esistenza della tag cw_mo_, che un oggetto di business potrebbe utilizzare per indicare qualsiasi meta oggetto secondario che il connettore utilizza. Il gestore dati ignora qualsiasi attributo identificato dalla tag cw_mo_ nelle informazioni specifiche dell'applicazione dell'oggetto di business. Per ulteriori informazioni sulla tag cw_mo_, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.

Utilizzo delle definizioni degli oggetti di business

Il gestore dati NameValue può convertire qualsiasi oggetto di business in dati serializzati NameValue sempre che l'oggetto di business fornisca dati in un formato che ottemperi ai requisiti del gestore dati. Il gestore dati NameValue richiede ogni parte dei dati per ottenere un nome che lo identifichi, come ad esempio BusinessObject=Customer, Verb=Create e CustomerName=JDoe. Siccome è necessario che gli attributi abbiano tale nome, è possibile che siano utilizzati con il gestore dati NameValue.

Sebbene gli oggetti di business esistenti che soddisfano tale requisito posso essere convertiti dal gestore dati NameValue, è buona norma creare i propri oggetti di

business per ciascun tipo di dati da elaborare. Se si utilizza un oggetto di business di esempio o un oggetto di business sviluppato per supportare la stessa applicazione in un'altra implementazione, assicurarsi di modificare la definizione il necessario per includere solo gli attributi richiesti per l'implementazione per cui si sta sviluppando.

Perciò, per convertire oggetti di business esistenti in una forma che corrisponda dettagliatamente ai propri dati, modificare l'oggetto di business per fornire solo i dati richiesti dall'applicazione e le informazioni richieste dal gestore dati. Per adattare oggetti di business esistenti per l'utilizzo con il gestore dati NameValue, effettuare le seguenti operazioni:

1. Effettuare un'analisi funzionale dell'applicazione selezionata e paragonare i risultati con gli oggetti di business esistenti per determinare i campi richiesti di una definizione oggetto di business.
2. Utilizzare Business Object Designer Express per aggiungere o eliminare, secondo necessità, attributi dalla definizione dell'oggetto di business.

Conversione degli oggetti di business in dati NameValue

Per convertire un oggetto di business in una stringa o flusso, il gestore dati NameValue effettua un loop tra gli attributi dell'oggetto di business in ordine sequenziale. Genera coppie nome-valore in maniera ricorrente nell'ordine in cui gli attributi si trovano nell'oggetto di business e nei relativi elementi secondari. Il nome dell'oggetto di business viene passato come un argomento al metodo di conversione.

Il gestore dati NameValue elabora gli oggetti di business nei dati NameValue come segue:

1. Il gestore dati crea una stringa per contenere i dati nell'oggetto di business.
2. Per determinare il nome dell'oggetto di business, il gestore dati aggiunge `BusinessObject=`*`Name`* alla stringa.
3. Per specificare il verbo, il gestore dati aggiunge `Verb=`*`Verb`* alla stringa.
4. Se l'attributo del meta oggetto `ValidateAttrCount` è impostato su `true`, il gestore dati aggiunge `AttributeCount=`*`Count`* alla stringa. Tale coppia nome-valore specifica il numero degli attributi nei dati dell'oggetto di business.
5. Il gestore dati esamina le informazioni dell'applicazione nella definizione dell'oggetto di business per determinare se vi siano meta oggetti secondari (i cui nomi sono elencati nella tag `cw_mo_` delle informazioni dell'applicazione dell'oggetto di business). Il gestore dati *non* include tali attributi nei dati NameValue. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
6. Il gestore dati effettua un loop tra i rimanenti attributi in ordine dell'oggetto di business, aggiungendo coppie nome-valore per ogni attributo semplice alla stringa. Per gli attributi del contenitore, il gestore dati effettua le seguenti operazioni:
   - Se l'attributo è un contenitore a cardinalità 1, il gestore dati aggiunge il nome dell'attributo ed un conteggio di 1 alla stringa e quindi elabora in maniera ricorrente l'oggetto di business secondario per aggiungere coppie nome-valore per ogni attributo nella stringa.
   - Se l'attributo è un contenitore a cardinalità n, il gestore dati aggiunge il nome dell'attributo e il numero di oggetti secondari nel contenitore alla stringa e quindi elabora in modo ricorrente ogni oggetto di business secondario, aggiungendo coppie nome-valore per ogni attributo nella stringa.

7. Quando il gestore dati completa la conversione, restituisce i dati serializzati al richiedente. Il gestore dati restituisce i dati nel formato (stringa o InputStream) desiderato dal richiedente.

Se l'attributo del meta oggetto secondario `ValidateAttrCount` è `true`, il gestore dati aggiunge un token contenente un numero di attributi nell'oggetto di business ai dati in uscita. Il gestore dati aggiunge il ritorno a capo ai dati in uscita; Il risultato finale è simile a Figura 36 su 162.

Conversione dei dati NameValue in oggetti di business

Questa sezione fornisce le seguenti informazioni su come il gestore dati NameValue converte le stringhe o i flussi in formati coppie nome-valore in oggetti di business:

- "Requisiti della stringa NameValue"
- "Elaborazione dati serializzati" a pagina 162

Requisiti della stringa NameValue

Il gestore dati NameValue crea i seguenti presupposti riguardo ai dati serializzati:

- Il nome dell'oggetto di business si trova nella prima coppia nome-valore.
- Il verbo si trova nella seconda coppia nome-valore.
- I dati contengono un token che rappresenta il numero di istanze degli oggetti secondari per ogni oggetto secondario contenuto in un oggetto di business.
- L'attributo `ObjectEventId` è presente in ogni oggetto di business.

Un token che rappresenta il numero degli attributi è facoltativo. Se l'attributo del meta oggetto secondario `ValidateAttrCount` è `true`, il gestore dati ricerca un token contenente un numero di attributi nell'oggetto di business. Se il numero degli attributi è specificato, è necessario che rifletta accuratamente il numero degli attributi nella definizione dell'oggetto di business.

Quando il gestore dati NameValue legge un file nel formato nome-valore, effettua i seguenti passaggi speciali di elaborazione per assegnare ad un attributo dell'oggetto di business il valore dell'attributo `CxIgnore` o `CxBlank`:

- Il gestore dati assegna la costante `CxIgnore` (`nulla`) come il corrispondente valore dell'attributo qualora incontri una qualsiasi delle seguenti condizioni nei dati NameValue:
  - Il valore della proprietà di configurazione dell'attributo del meta oggetto `CxIgnore` (il valore configurato per la relativa proprietà Valore predefinito)
  - Il valore di una stringa vuota (" ")
  - Nessun valore corrispondente nei dati NameValue per l'attributo dell'oggetto di business.
- Il gestore dati assegna la costante `CxBlank` come il valore dell'attributo solo quando l'attributo del meta oggetto `CxBlank` è configurato ed incontra tale valore configurato nei corrispondenti dati NameValue.

Figura 36 mostra un esempio di dati serializzati nel formato NameValue.

```
BusinessObject=Customer
  Verb=Update
          AttributeCount=7
          CustomerID=103
          CustomerName=Thai Inc.
          Cust_Phone_Number=CxIgnore
          ProductName=GoodProduct
          Address=2
            BusinessObject=Address
              Verb=Update
              AttributeCount=3
              AddressID=105
              AddressLine=CxIgnore
              ObjectEventID=12345
            BusinessObject=Address
              Verb=Delete
              AttributeCount=3
              AddressID=106
              AddressLine=2758 Forest Avenue
              ObjectEventID=CxIgnore
          Item=1
            BusinessObject=Item
              Verb=Update
              ItemID=107
              ItemName=CxIgnore
              ObjectEventID=Obj_201
          ObjectEventID=SampleConnector_894927711_2
```

Figura 36. Esempio di dati NameValue

In tale esempio, le voci indicano:

- `BusinessObject` è il nome dell'oggetto di business principale o secondario che viene processato.
- `Verb` è il tipo di richiesta (ad esempio, Create o Update) con cui l'oggetto di business principale o secondario che viene inviato.
- `AttributeCount` è il numero totale di attributi per l'oggetto di business principale o secondario a quel livello.
- `CustomerID`, `CustomerName`, `Cust_Phone_Number` e `ProductName` sono i nomi degli attributi per l'oggetto di business superiore. I valori per ogni attributo dell'oggetto di business principale seguono il nome dell'attributo.
- `Address = 2` indica la presenza di due istanze dell'oggetto di business secondario `Address`. `Address` è il nome dell'attributo che si riferisce all'oggetto di business secondario `Address` nell'oggetto principale.
- `Item = 1` indica che l'attributo Item contiene un'istanza singola dell'oggetto di business `Item`.
- `AddressID` e `AddressLine` sono i nomi degli attributi per l'oggetto di business secondario `Address`. I valori per ogni attributo dell'oggetto di business secondario seguono il nome dell'attribute.
- `ObjectEventID=Obj_201` è l'ID generata dal sistema per l'oggetto di business secondario, `Item`.
- `ObjectEventID=SampleConnector_894927711_2` è l'ID generata dal sistema per l'oggetto di business principale, `Customer`.

Elaborazione dati serializzati

Il gestore nome NameValue converte stringhe o flussi in formato coppie nome-valore in un oggetto di business come di seguito indicato:

1. Il gestore dati crea un oggetto di business per contenere i dati nella stringa o nel flusso.

2. Il gestore dati imposta il verbo nell'oggetto di business. Il gestore dati suppone che il verbo per l'oggetto di business di livello superiore è nella seconda coppia nome-valore nei dati. Notare che gli oggetti di business secondari possono non avere verbi impostati.
3. Se l'attributo del meta oggetto secondario `ValidateAttrCount` è impostata su `true`, il gestore dati verifica che il numero degli attributi nel file corrisponda al numero di attributi nella definizione dell'oggetto di business.
4. Il gestore dati analizza i dati serializzati.
   - Prima di tutto determina se vi siano meta oggetti secondari (quelli i cui nomi sono elencati nella tag `cw_mo_` delle informazioni dell'applicazione dell'oggetto di business). Il gestore dati *non* effettua l'elaborazione per popolare questi attributi dell'oggetto di business. Per ulteriori informazioni sulla tag `cw_mo_`, fare riferimento a "Implementazione della conversione da un oggetto di business" a pagina 186.
   - Immette i valori dei rimanenti attributi semplici nell'oggetto di business. Il gestore dati elabora gli attributi del contenitore come segue:

     Se l'attributo è a cardinalità singola, il gestore dati analizza in modo ricorrente gli attributi nell'elenco dell'attributo e aggiunge il contenitore dell'oggetto di business secondario all'oggetto di business principale.

     Se l'attributo è a cardinalità multipla, il gestore dati analizza in modo ricorrente gli attributi in ogni elenco dell'attributo secondario e aggiunge il contenitore dell'oggetto di business all'oggetto di business principale.

E' possibile specificare gli attributi nei dati serializzati in qualsiasi ordine per le conversioni stringa in oggetto di business poiché il gestore dati crea un nome ed un'associazione del valore.

Capitolo 9. Gestore dati Complex Data

Il gestore dati per Complex Data IBM WebSphere Business Integration Server Express, chiamato *gestore dati Complex Data*, converte gli oggetti di business in formati dati binari o di testo e converte tali formati di nuovo in oggetti di business. Il gestore dati Complex Data è in grado di convertire un numero quasi infinito di formati proprietari ed ereditari in oggetti di business e viceversa. I formati supportati includono documenti MS Office, PDF, COBOL Copybooks, tutti i formati basati su XML, tutti i formati di testo (delimitati o fixedwidth) e molti altri.

Nota: Il gestore dati Data Complex non è disponibile per l'utilizzo con Linux o OS/400 e i5/OS.

Questo capitolo descrive come il gestore dati Complex Data elabora i formati di dati e come generare definizioni degli oggetti di business per l'utilizzo con i gestori dati. E' possibile utilizzare queste informazioni come una guida per implementare una soluzione che influenzi le capacità di conversione dati del gestore dati Complex Data. Questo capitolo descrive inoltre come configurare il gestore dati Complex Data.

Questo capitolo contiene le sezioni seguenti:

Panoramica

Il gestore dati Complex Data è un modulo di conversione dati il cui ruolo principale è convertire gli oggetti di business in specifici formati di dati e viceversa. Per effettuare tali conversioni il gestore dati Complex Data utilizza il ContentMaster prodotto da Itemfield, Inc., così come IBM WebSphere Business Integration Server Express XML Data Handler.

ContentMaster converte uno specificato formato di dati binario o di testo in XML che può quindi essere convertito in oggetti di business dal gestore dati XML. Per ulteriori informazioni riguardo i componenti coinvolti in questa conversione fare riferimento a "Componenti di esecuzione" a pagina 166, fare riferimento anche a "Convertire oggetti di business in formati dati specificati" a pagina 169 e "Conversione di formati dati specificati in oggetti di business" a pagina 170 per informazioni riguardo a come tali componenti interagiscono durante l'esecuzione.

Componenti gestore dati Complex Data

ContentMaster fornisce sia i componenti di sviluppo che i componenti di esecuzione. I componenti di sviluppo sono utilizzati per configurare i componenti di esecuzione e per creare gli schemi e i parser necessari per effettuare le conversioni durante l'esecuzione. Le seguenti due sezioni illustrano come gli ambienti ContentMaster di sviluppo ed esecuzione interagiscono con i componenti di un adattatore esistente per formare il gestore dati per Complex Data. Per

ulteriori informazioni sui componenti di sviluppo ed esecuzione ContentMaster, fare riferimento alla documentazione di Itemfield ContentMaster.

Componenti di sviluppo

I componenti di sviluppo sono i seguenti:

- Itemfield ContentMaster Studio (CMStudio)
- Business Object Designer Express
- XML Object Discovery Agent

L'interazione tra questi componenti è la seguente:

- Un file contenente dati immessi è caricato nell'applicazione CMStudio. L'utente utilizza CMStudio per creare un Parser e/o un Serializer per il formato di immissione dato, oltre a un file definizione schema XML (.xsd).
- L'utente pubblica il Parser e il Serializer nell'ambiente di esecuzione di ContentMaster ed esporta il file definizione schema XML da CMStudio.
- Il file definizione schema XML viene utilizzato come una definizione schema di immissione per l'ODA XML. Fare riferimento a "Utilizzare l'ODA XML" per ulteriori informazioni sull'utilizzo dell'ODA XML.
- ODA XML genera una definizione dell'oggetto di business che può essere memorizzata nel magazzino di InterChange Server Express e può essere utilizzata per generare oggetti di business.

Appena la definizione di oggetto di business è stata creata e ContentMaster Engine (CMEngine) è stato correttamente configurato e pubblicato, l'adattatore può convertire dinamicamente dati di immissione dall'adattatore agli oggetti di business e gli oggetti di business di nuovo in dati che l'adattatore può elaborare. La sezione seguente descrive come questo processo funziona durante l'esecuzione.

Componenti di esecuzione

I componenti di esecuzione sono i seguenti:

- WebSphere Business Integration Server Express Adapter esistente
- Gestore dati Complex Data
- ContentMaster Engine
- gestore dati XML

L'ambiente di esecuzione comprende *notifica degli eventi*, in cui i dati immessi che si originano all'esterno di InterChange Server Express sono convertiti da un adattatore e inviati a InterChange Server Express, *elaborazione delle richieste*, in cui InterChange Server Express utilizza l'adattatore per convertire oggetti di business e invia i dati risultanti all'applicazione, al file, o ad altri clienti di questi dati. In entrambi i casi, l'adattatore chiama il gestore dati Complex Data per effettuare la conversione tra i dati e XML e chiama il gestore dati XML per convertire i dati tra XML e gli oggetti di business.

Fare riferimento a "Conversione di formati dati specificati in oggetti di business" a pagina 170 per ulteriori dettagli su gestione degli eventi durante l'esecuzione e fare riferimento a "Convertire oggetti di business in formati dati specificati" a pagina 169 per ulteriori informazioni sul processo di elaborazione delle richieste.

Configurazione del gestore dati Complex Data

Per configurare il gestore dati Complex Data da utilizzare con un connettore, eseguire i seguenti passi:

- Installare, registrare e verificare ContentMaster
- Configurare il gestore dati Complex Data per l'utilizzo con un adattatore
- Configurare il meta oggetto del gestore dati Complex Data

Ciascuno dei seguenti passi è descritto più dettagliatamente nelle sezioni seguenti.

Installare, registrare e verificare ContentMaster

Per installare ContentMaster, seguire le istruzioni per l'installazione fornite nella documentazione Itemfield ContentMaster per la propria versione di ContentMaster e per il proprio sistema operativo. Tale documentazione fornisce requisiti hardware e software così come istruzioni per l'installazione, la registrazione e la verifica di ContentMaster.

Quando viene richiesto dal programma di installazione di ContentMaster per l' ubicazione Java Run-time Environment (JRE), assicurarsi di selezionare JRE rilasciato con gli adattatori WBI. WBI JRE è ubicato nella directory `<WBI install dir>/jre/bin/classic`.

Nota: La riga di comando dell'installazione di ContentMaster non richiede l'ubicazione JRE. Nel caso in cui si desideri modificare manualmente l'ubicazione JRE dopo l'installazione, consultare la documentazione ContentMaster relativa alla versione e alla piattaforma in uso per determinare quali file o variabili di ambiente devono essere modificate per specificare un'ubicazione JRE differente.

Infine, registrare e verificare l'installazione di ContentMaster seguendo le indicazioni nella documentazione di ContentMaster prima di procedere con la configurazione dell'adattatore.

Configurazione del gestore dati per l'utilizzo con l'adattatore

Per utilizzare il gestore dati Complex Data con un adattatore WebSphere Business Integration Server Express, è necessario modificare lo script di avvio dell'adattatore per includere i file richiesti dal gestore dati Complex Data. Le modifiche richieste per lo script di avvio del connettore sono elencate di seguito.

- Creare una variabile denominata `CMDir` e impostare il suo valore all'ubicazione dell'installazione di ContentMaster.

  Esempio Windows:

  ```
  impostare CMDIR = "C:\Program Files\Itemfield\ContentMaster"
  ```

- Creare una variabile denominata `CDDHJars`, impostare il suo valore all'ubicazione di BIA_CwComplexDatahHandler.jar e CMJava_API.jar.

  Esempio Windows:

  ```
  impostare CDDHJars =
  %CROSSWORLDS%\DataHandlers\BIA_CwComplexDataHandler.jar;
  %CMDIR%\CMJava_API.jar
  ```

- Modificare la riga di esecuzione java esistente in modo da includere la variabile `CMDIR` nel percorso libreria e la variabile `CDDHJars` nel percorso classe.

  Esempio Windows:

```
%CWJAVA% ...
-Djava.library.path="%CMDIR%";...
-cp "%CDDHJars%";...
```

Configurazione del meta oggetto di configurazione del gestore dati Complex Data

Per configurare il gestore dati Complex Data, è necessario assicurarsi che le sue informazioni di configurazione siano fornite nel gestore dati Complex Data meta oggetto secondario.

Per il gestore dati Complex Data, IBM fornisce il meta oggetto secondario predefinito `MO_DataHandler_Complex`. Ogni attributo in questo meta oggetto definisce una proprietà di configurazione per il gestore dati Complex Data. Il meta oggetto secondario predefinito per il gestore dati Complex Data contiene cinque attributi, uno dei quali è un riferimento al meta oggetto di definizione del gestore dati XML. Tali attributi sono definiti in Tabella 64.

Tabella 64. Attributi del meta oggetto secondario per il gestore dati Complex Data

Nome attributo	Descrizione	Valore predefinito fornito
`ClassName`	Nome della classe gestore dati da caricare per l'utilizzo con il MIME type specificato. Il meta oggetto superiore del gestore dati ha un attributo il cui nome corrisponde al MIME type specificato e il cui tipo è il meta oggetto secondario Complex Data.	`com.ibm.adapters. datahandlers.complex. Complex`
`CMParser`	Nome sotto il quale è pubblicato Content Master Parser.	*Nessuno*
`CMSerializer`	Nome sotto il quale ContentMaster Serializer è pubblicato.	*Nessuno*
`ObjectEventId`	Attributo Placeholder richiesto dal sistema di integrazione business.	*Nessuno*
`XMLDH_ConfigMO`	Il meta oggetto secondario per il gestore dati XML che il gestore dati Complex Data utilizza.	`MO_DataHandler_XML`

Il "Valore predefinito fornito" nella colonna Tabella 12 elenca il valore nelle proprietà del Valore predefinito per l'attributo corrispondente negli oggetti di business forniti. E' necessario verificare il proprio ambiente e impostare le proprietà del Valore predefinito di tali attributi con i valori appropriati. E' necessario assicurarsi che tutti gli attributi abbiano valori predefiniti eccetto dove la conversione sarà unidirezionale. In questo caso è necessario specificare o un valore `CMParser` o un valore `CMSerializer`.

Nota: Utilizzare Business Object Designer Express per modificare le definizioni degli oggetti di business.

Un singolo meta oggetto secondario può specificare solo una coppia Parser/Serializer. Se il proprio connettore richiede l'elaborazione di formati di dati multipli, è necessario creare meta oggetti secondari separati per ciascuna coppia Parser/Serializer. Per creare configurazioni multiple del gestore dati Complex Data, procedere nel modo seguente:

- Copiare e rinominare il meta oggetto secondario predefinito Complex Data.
- Impostare i valori predefiniti degli attributi per ciascun meta oggetto secondario Complex Data in modo da configurare l'istanza del gestore dati.
- Associare i meta oggetti secondari con i dati di immissione.

Nota: L'associazione di questi meta oggetti secondari con i dati di immissione è specifica per ciascun adattatore. Consultare la documentazione adattatore appropriata per informazioni su come associare ognuno dei meta oggetti secondari con i formati dati di immissione appropriati.

Per ulteriori informazioni su come configurare un gestore dati, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

Convertire oggetti di business in formati dati specificati

Per convertire un oggetto di business in un formato dati specificato, il gestore dati Complex Data richiama il gestore dati XML per convertire l'oggetto di business ricevuto in XML. Il gestore dati quindi richiama il ContentMaster Serializer specificato dal meta oggetto secondario del gestore dati associato per convertire XML nel formato dati specificato.

Figura 37 illustra l'elaborazione delle richieste con il gestore dati Complex Data:

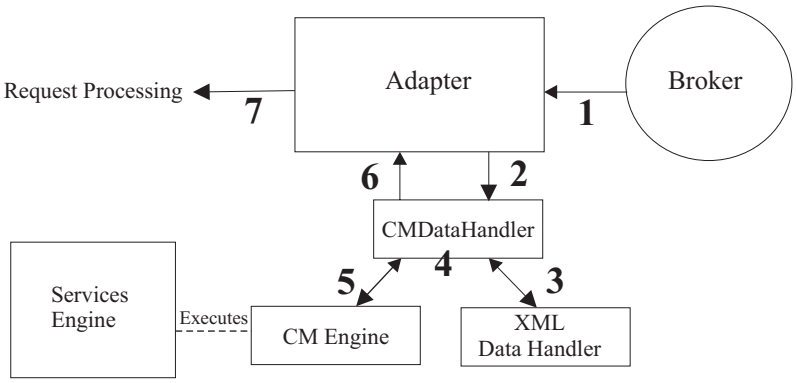

Figura 37. Elaborazione richieste con il gestore dati Complex Data

1. L'adattatore riceve una richiesta di servizio.
2. L'adattatore invia l'oggetto di business al gestore dati Complex Data utilizzando i metodi `getStringFromBO()` o `getStreamFromBO()` dal gestore dati API.
3. Il gestore dati Complex Data richiama il gestore dati XML, trasferendo l'oggetto di business. Il risultato dell'esecuzione è una stringa XML.
4. Il gestore dati Complex Data riceve la stringa XML dal gestore dati XML e inizializza una sessione con ContentMaster engine.
5. ContentMaster engine è richiamato con la stringa XML e con il nome del Serializer determinato dal meta oggetto di configurazione del gestore dati. L'output del Content Master engine può essere sia una stringa che un byte[].
6. Il gestore dati Complex Data invia sia una stringa(se testo) che un InputStream(se byte[]) all'adattatore richiedente.
7. I dati nativi sono inviati ad un'applicazione, ad un file, ad un trasporto, o ad un altro client basato su l'adattatore utilizzato e la sua configurazione.

Conversione di formati dati specificati in oggetti di business

Per convertire un formato dati specifico in un oggetto di business, il gestore dati Complex Data richiama il Parser ContentMaster specificato dal meta oggetto secondario del gestore dati associato. Utilizza il gestore dati associato per convertire il formato dati specificato in XML. Il gestore dati quindi richiama il gestore dati XML per convertire questo XML in un oggetto di business.

Figura 38 illustra l'evento gestione del processo di esecuzione ContentMaster.

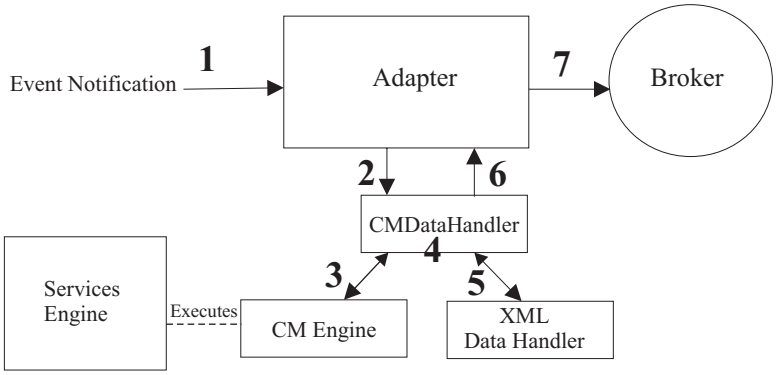

Figura 38. Notificazione evento con il gestore dati Complex Data

1. L'adattatore riceve un evento di notificazione.
2. L'adattatore invia i dati al gestore dati Complex Data.
3. Il gestore dati Complex Data richiama ContentMaster Engine, trasferendo i dati di immissione e il nome del parser da eseguire. Il parser esegue l'operazione e restituisce la stringa XML.
4. Il gestore dati Complex Data data determina il nome dell'oggetto di business e il verbo associato.

   La risoluzione nome è eseguita da NameHandler specificato nel meta oggetto di configurazione del gestore dati XML. Il meta oggetto di configurazione del gestore dati è specificato dall'attributo `XMLDH_ConfigMO` del meta oggetto gestore dati Complex Data.

   La risoluzione del verbo è eseguita in base al processo specificato nella seguente sezione Risoluzione verbo.
5. Il gestore dati Complex Data utilizza il gestore dati XML per elaborare la stringa XML e l'istanza oggetto di business dal passo 3.
6. L'oggetto di business compilato creato dalla stringa XML viene restituito all'adattatore richiedente.
7. L'oggetto di business viene trasferito su InterChange Server Express.

Risoluzione verbo

Durante la conversione da uno specifico formato dati ad un oggetto di business, all'oggetto di business può essere assegnato un verbo. Tale processo viene detto una risoluzione verbo. Esistono due modi in cui un verbo può essere impostato su un'istanza oggetto di business. La risoluzione verbo viene eseguita nel seguente ordine:

1. Il gestore dati Complex Data controlla XML ricevuto da ContentMaster per un'istruzione di elaborazione nel formato: `<?Verb Create?>`

2. Il gestore dati Complex Data controlla il campo informazioni specifiche dell'applicazione di Business Object Level Application per la stringa: Verbo=Crea. (Notare che: con tale approccio BO sarà assegnato ad uno specifico verbo)

Se il verbo non può essere determinato utilizzando uno degli schemi sopra elencati, il verbo rimarrà non impostato (`CxBlank`). Se il verbo non è impostato, si suppone che l'adattatore richiedente imposterà il verbo.

Gestione degli errori

Tabella 65 descrive gli errori che si presentano nei vari componenti di un sistema gestore dati Complex Data e come dovrebbero essere gestiti.

Tabella 65. Gestione degli errori nel gestore dati Complex Data

ContentMaster Engine	Se si verifica un errore durante l'esecuzione di ContentMaster Engine, viene registrata un'eccezione nel gestore dati Complex Data. La stringa eccezione contiene: • Dettagli sull'eccezione elaborata. • Un URL indica un file che, una volta caricato in CMStudio, descrive lo specifico errore parser o serializer.
gestore dati XML	Per ulteriori informazioni sulla gestione degli errori nel gestore dati XML, fare riferimento a Capitolo 3, "Gestore dati XML", a pagina 29.
ODA XML	Per ulteriori informazioni sulla gestione degli errori nell'ODA XML, fare riferimento a "Utilizzare l'ODA XML", a pagina 217.
gestore dati Complex Data	Qualora dovesse verificarsi un errore in uno qualsiasi dei componenti sopra citati, l'elaborazione si interrompe e un'eccezione viene inviata all'agente connettore richiedente. Ciascun errore contiene un testo descrittivo che sarà visibile nel file di log Connector Agent.

Parte 2. Gestori dati personalizzati

Capitolo 10. Configurazione di un gestore dati personalizzato

Questo capitolo fornisce informazioni su come implementare un gestore dati personalizzato per l'utilizzo con un adattatore WebSphere Business Integration Server Express o con un client di accesso se si utilizza InterChange Server Express. Così come per gestori dati IBM, un gestore dati personalizzato converte un oggetto business in uno specifico formato dati serializzati e converte i dati serializzati in uno specifico formato in un oggetto business. Questo capitolo contiene le sezioni seguenti:

Panoramica del processo di sviluppo del gestore dati

Per sviluppare un gestore dati personalizzato, si codifica il file sorgente del gestore dati e si eseguono le altre attività, come ad esempio sviluppare un meta oggetto per il gestore dati. L'attività di creazione di un gestore dati personalizzato include i seguenti passaggi generali:

1. Progettare il gestore dati, basato sul formato dei dati serializzati e sulla struttura degli oggetti business da convertire.
2. Creare una classe che estenda la classe:

   `com.crossworlds.DataHandlers.DataHandler`
3. Implementare i metodi astratti che convertono i dati tra formati dati specifici e oggetti business.
4. Compilare la classe e aggiungerla al file `CustDataHandler.jar`.
5. Creare i meta-oggetti del gestore dati.
6. Sviluppare le definizioni oggetti business in conformità con i requisiti del gestore dati così come con i requisiti del richiedente.

Figura 39 fornisce una panoramica visiva del processo di sviluppo del gestore dati e e fornisce un rapido riferimento ai capitoli in cui è possibile trovare le informazioni sugli argomenti specifici. Notare che, se un team è disponibile per lo sviluppo del gestore dati, le principali attività di sviluppo del gestore dati possono essere svolte in parallelo con differenti membri del team di sviluppo.

Figura 39. Panoramica sul processo di sviluppo del gestore dati

Strumenti per lo sviluppo del gestore dati

Poiché i gestori dati sono scritti in Java, questi possono essere sviluppati sia su sistemi Windows che Linux. Tabella 66 elenca gli strumenti che IBM fornisce per lo sviluppo di gestori dati.

Tabella 66. Strumenti per lo sviluppo del gestore dati

Strumento di sviluppo	Descrizione
Kit di sviluppo adattatore (ADK)	Include quanto segue: • Gestori dati di esempio • File matrice per la classe estesa `DataHandler`
Gestore dati API	Classe singola, `DataHandler`, che è possibile estendere per creare un gestore dati personalizzato.
Kit di sviluppo del conettore Java (JCDK)	Contiene classi Java per operare con oggetti business.

Il kit di sviluppo adattatore

Il Kit di sviluppo adattatore (ADK) è un prodotto separato (non parte di WebSphere Business Integration Server Express) che fornisce file ed esempi che assistono l'utente nello sviluppo di un adattatore. Fornisce, inoltre, esempi per molti dei componenti dell'adattatore, incluso ODA (Object Discovery Agent), un connettore e un gestore dati. ADK fornisce questi esempi nella sottodirectory DevelopmentKits della directory del prodotto.

Nota: ADK è parte del prodotto WebSphere Business Integration Adapters e richiede il suo programma di installazione. Perciò, per avere accesso agli esempi di sviluppo in ADK, è necessario acquistare ed installare il prodotto WebSphere Business Integration Adapter Framework ed installare l'ADK. Si noti che ADK è disponibile soltanto per sistemi Windows.

Tabella 67 elenca gli esempi che ADK fornisce per lo sviluppo di un gestore dati oltre alla subdirectory della directory DevelopmentKits in cui risiedono.

Tabella 67. Esempi ADK per lo sviluppo di gestori dati

componente Kit sviluppo adattatore	subdirectory DevelopmentKits
Esempi gestori dati	`edk\DataHandler`

Esempi gestori dati

Per l'assistenza allo sviluppo di un gestore dati, ADK include un codice per numerosi esempi di gestori dati nella seguente directory del prodotto:

`DevelopmentKits\edk\DataHandler\Samples`

Tabella 68 elenca il gestore dati d'esempio che ADK fornisce.

Tabella 68. Gestori dati di esempio inclusi in EDK

Nome	Descrizione
`delimited.java`	Converte oggetti business in Stringhe delimitate e stringhe delimitate in oggetti business.
`fixedwidth.java`	Converte oggetti business in Stringhe FixedWidth e da stringhe FixedWidth in oggetti business.
`namevalue.java`	Converte oggetti business in Stringhe NameValue e stringhe NameValue in oggetti business.

Nota: Mentre tali esempi sono utili da esaminare, essi *non* forniscono esempi di tutte le funzionalità supportate nella classe `DataHandler`.

Sviluppo file

La directory `DevelopmentKits\edk\DataHandler` fornisce numerosi file per l'assistenza nello sviluppo di un gestore dati personalizzato, inclusi quelli elencati in Tabella 69.

Tabella 69. File di sviluppo gestore dati

file di sviluppo gestore dati	Per ulteriori informazioni
`StubDataHandler.java`	"Estensione della classe base del gestore dati" a pagina 179
`makeDataHandler.bat` (sistemi Windows)	"Inserimento di un gestore dati nel file jar" a pagina 195

Gestore dati API

Il gestore dati API fornisce una singola classe Java, chiamata `DataHandler`. La classe di base astratta `DataHandler` semplifica lo sviluppo di un gestore dati personalizzato. Tale classe contiene i metodi che compilanoono un oggetto business con valori estratti da dati di immissione e metodi che serializzano un oggetto business in una stringa o flusso. La classe inoltre include metodi di utilità che un gestore dati personalizzato può utilizzare. E' possibile far derivare un gestore dati personalizzato dalla classe `DataHandler`.

Per ulteriori informazioni sui metodi nella classe `DataHandler`, fare riferimento a Capitolo 11, "Metodi della classe base del gestore dati", a pagina 203.

Java Connector Development Kit

Per lavorare con oggetti di business, un gestore dati deve utilizzare metodi dalle classi nel ava Connector Development Kit (JCDK). Sviluppando il gestore dati potrebbe rendersi necessario importare ulteriori classi JCDK, come `CxCommon.CxObjectContainerInterface` oppure `CxCommon.CxObjectAttr`.

Nota: Il JCDK rappresenta la libreria del connettore Java di livello inferiore. .

Progettazione del gestore dati

Prima di iniziare a scrivere un gestore dati personalizzato, è necessario conoscere:

- Il formato dei dati dei file che il gestore dati convertirà
- Il modello di oggetto di business

  In particolare, è necessario saper:

– estrarre valori da un'istanza di oggetto di business
– creare un'istanza di oggetto di business ed associarvi dei valori

Creazione di un gestore dati con metadati

Affinchè il gestore dati sia con metadati, esso deve specificare dinamicamente le informazioni che identificano il gestore dati da utilizzare. Per ulteriori informazioni sul gestore dati con metadati, consultare"Progettazione del gestore dati con i metadati" a pagina 20.

Utilizzo dei meta-oggetti del gestore dati

in fase di progettazione è ncessario stabilire se il gestore dati utilizzerà o meno meta-oggetti per inizializzare la configurazione.

Nota: Per ulteriori informazioni sui meta oggetti, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

Prima di decidere se utilizzare o meno meta-oggetti bisogna tener presente quanto segue:

- I meta-oggetti permettono una configurazione dinamica del gestore dati. Questa strategia di progettazione crea un gestore dati più flessibile che può essere configurato in base al contesto in cui viene richiamato.

  To call a data handler that uses meta-objects, the caller passes the data handler's associated MIME type into the `metodo createHandler()`. Se richiamato con un tipo MIME, il metodo `createHandler()` inizializza un gestore dati appena istanziato con le informazioni di configurazione nel meta-oggetto secondario
- Si verifica un sovraccarico associato all'accesso ed alla ricerca di meta-oggetti. Il gestore dati potrebbe non voler utilizzare meta-oggetti se le informazioni di configurazione non vengono modificate (perchè codificate) ovvero se converte dati a cui non è stato associato un tipo MIME.

  Per richiamare un gestore dati che *non* utilizza meta-oggetti, il richiedente invia *solo* il nome classe del gestore dati al `metodo createHandler()`. Se richiamato con un nome classe, `createHandler()` istanzia un gestore dati di quel nome class senza ricercare meta-oggetti associati

Se si progetta un gestore dati personalizzato che utilizzi meta-oggetti, occorre create tali meta-oggetti come parte dell'implementazione del gestore dati. Per ulteriori informazioni, consultare "Creazione dei meta-oggetti di un gestore dati" a pagina 196.

Estensione della classe base del gestore dati

Per estendere un gestore dati personalizzato, bisogna estendere il classe di base del gestore dati`DataHandler`) per creare il proprio*classe del gestore dati*. La classe `DataHandler` contiene metodi che effettuano operazioni di conversione class (stringa ad oggetto di business ed oggetto di business a stringa, e metodi di utilità che forniscono assistenza nello sviluppo . Il EDK contiene codici matrice e file di esecuzione per un gestore dati personalizzato. Il file matrice contiene un codice Java che definisce una classe vuota con un elenco di tutti i metodi da implementare. Si può utilizzare il file matrice come una maschera per generare un gestore dati personalizzato.

Per creare un file sorgente del gestore dati utilizzando il file matrice

1. Copiare il file StubDataHandler.java e rinominarlo in modo che il nome del file corrisponda al nome della classe del gestore dati da esso definita.

   Il file matrice risiede nella sottodirectory DevelopmentKits\edk\DataHandler nella directory del prodotto. Esso include istruzioni di import che importanto il pacchetto del gestore daticom.crossworlds.DataHandlers. Inoltre importa alcune classi dal Java Connector Development Kit.
2. Sostituire la parola chiave dello StubDataHandler con il nome della classe che implementa il gestore dati personalizzato.

   As esempio, la riga seguente estende la classe DataHandler per creare una classe del gestore dati personalizzato chiamata class to create a custom HtmlDataHandler :

   ```
   classe pubblica di HtmlDataHandler extends DataHandler
   ```

Implementazione dei metodi

Per sviluppare un gestore dati è necessario implementare i seguenti metodi della classe DataHandler :

- I metodi astratti del DataHandler (richiesti)
- I metodi pubblici del DataHandler (facoltativi)

I metodi del gestore dati personalizzato vanno nel file sorgente Java della classe DataHandler che è stata creata "Estensione della classe base del gestore dati" a pagina 179.

Nota: Se il richiedente riutilizza un istanza memorizzata nella cache della classe DataHandler su thread multipli, potrebbe essere necessario rendere la classe protetta dai thread.

Implementazione dei metodi astratti

La classe di base del gestore dati, DataHandler, fornisce i metodi Tabella 70, che occorre *necessariamente* implementare nel DataHandler classe per il gestore dati personalizzato

Tabella 70. Metodi membri nella classe DataHandler

Conversione dei dati	Formato dei dati serializzati	Metodo DataHandler
Conversione stringa ad oggetto di business	Converte dati serializzati, a cui si accede tramite un oggetto Reader , in un oggetto di business	getBO() - abstract
Conversione oggetto di business a stringa	Converte un oggetto di business in un oggetto InputStream	getStreamFromBO()
	Converte un oggetto di business in un oggetto String	getStringFromBO()
	Converte un oggetto di business in un array di byte.	getByteArrayFromBO()

Nota: La copia del StubDataHandler.java file che estende la classe DataHandler con il gestore dati personalizzato contiene dichiarazioni per i metodi astratti che occorre implementare.

Le seguenti sezioni forniscono informazioni di implementazione per ciascun metodo astratto DataHandler .

Implementazione della conversione in oggetto di business

Il metodo astratto `metodo getBO()` effettua operazioni di conversione da stringa ad oggetto di business; ossia, compilaie un oggetto di business con dati estratti da un oggetto Java `Reader` . Esistono due versioni del metodo `getBO()` :

- `getBO(Reader serializedData, BusinessObjectInterface theObj, Object config)`

  Gli argomenti di input includono un oggetto `Reader` che contiene i dati serializzati ed un riferimento all'oggetto di business. Il metodo compila l'oggetto di business *theBusObj* con i dati *serializedData* .
- `getBO(Reader serializedData, Object config)`

  Gli argomenti di input includono un oggetto `Reader` che contiene i dati serializzati. Il metodo determina il nome del tipo di oggetto di business (definizione delloggetto di business) sulla base dei dati e poi crea e compila un'sistanza di oggetto di business di quel tipo.

Nota: Per supportare il gestore dati richiamato nel contesto di un connettore, la classe del gestore dati (che estende il `DataHandler`) deve implementare *entrambe* le versioni del metodo `getBO()` . Per supportare il gestore dati richiamato *solo* da un client di accesso(IBM WebSphere Business Integration Server Express, bisogna implementare solo il secondo modulo del metodo `getBO()` ; il Server Access Interface utilizza solo questo secondo modulo del `getBO()`.

Il metodo `getBO()` consente al richiedente di passare un oggetto facoltativo contenente informazioni di configurazione (il parametro *config*). Queste informazioni vanno ad aggiungersi alle informazioni di configurazione specificate nel meta-oggetto del gestore dati. Ad esempio,l'oggetto di configurazione può indicare un file maschera o un URL utilizzati dal gestore dati.

Nota: Durante la conversione in oggetto di business, il metodo `getBO()` deve assicurare che ogni attributo identificato con una tag `cw_mo_` *label* nelle informazioni specifiche sull'applicazione del principale l'oggetto di business *non* riceve alcun valore. Per ulteriori informazioni sulla tag `cw_mo_` *label* , consultare"Implementazione della conversione da un oggetto di business" a pagina 186.

Lo scopo del metodo astratto `getBO()` è di compilare un oggetto di business con i dati seralizzati contenuti nell'oggetto `Reader` . La versione pubblica del metodo `getBO()` può così ricevere i dati serializzati in una delle diverse forme supportate, convertire i dati in un oggetto Reader e poi richiamare il metodo astratto `getBO()` per realizzare la conversione da stringa ad oggetto di business. Per maggiori informazioni sul metodo pubblico `getBO()` , consultare "getBO() - public" a pagina 206.

Figura 40 mostra un'implementazione base del secondo modulo del metodo `getBO()` . Questo esempio illustra i passaggi nella conversione di dati da un oggetto `Reader` contenente dati a larghezza fissa in un oggetto di business:

1. Il metodo `getBO()` converte i dati nell'oggetto `Reader` in un oggetto `String` object. Successivamente esso richiama il metodo `getBOFromString()` definito dall'utente per creare un'istanza dell'oggetto di business.
2. Il metodo `getBOFromString()` stabilisce che tipo di oggetto di business creare in base al primo token a larghezza fissa nella `String` e poi crea un'istanza dell'oggetto di business per quel tipo. Esso riceve il verbo dal secondo token a larghezza fissa nella `String` ed imposta il verbo nell'oggetto di business. Questo

metodo poi richiama il metodo `parseAttributeList()` definito dall'utente per analizzare il resto della `String` e compilare l'oggetto di business con dei valori.

3. Il metodo `parseAttributeList()` analizza la `String` e compila ripetutamente l'oggetto di business. Quando il metodo trova un attributo di un tipo di oggetto stabilisce se l'oggetto è a cardinalità singola o multipla. Esso richiama il `getMultipleCard()` per elaborare ripetutamente l'oggetto di business nell'array e il `getSingleCard()` per elaborare un oggetto di business secondario a cardinalità singola.

Suggerimenti: Un gestore dati estrae dati da oggetti di business e compila oggetti di business con dati allo stesso modo di un connettore. Nel seguente esempio di codice, il metodo `getBOFromString()` richiama il `JavaConnectorUtil.createBusinessObject()` per creare un'istanza di oggetto di business e il `BusinessObjectInterface.setVerb()`per impostare il verbo.

```
BusinessObjectInterface getBO(Reader serializedData pubblico,
    Object config)
    throws Exception
{
    clear(config);
    BusinessObjectInterface resultObj = null;

    // Creare un oggetto Stringa dal Reader, poi utlizzare la stringa
    // metodo
    int conversionCheck;
    char[] temp = new char[2000];
    StringBuffer tempStringBuffer = new StringBuffer(1000);

    while ( (conversionCheck = serializedData.read(temp)) != -1 )
        tempStringBuffer.append(new String (temp, 0, conversionCheck));

    mBOString = new String(tempStringBuffer);
    mBOStringLength = mBOString.length();

    resultObj = getBOFromString(null);
    return resultObj;
}


// Ottiene il nome ed il verbo dell'oggetto di business e crea un'istanza dell'oggetto di business
BusinessObjectInterface getBOFromString(String pvBOType)privato
    throws Exception
{
    BusinessObjectInterface returnObj = null;
    String lvBOName = null;
    String lvVerb = null;

    lvBOName = this.getNextToken(mBONameSize, true);
    lvVerb = this.getNextToken(mBOVerbSize, true);

    if ( (pvBOType != null) && (lvBOName.compareTo(pvBOType) != 0) ) {
        throw new Exception(...);
    }
    else
    {
        returnObj = JavaConnectorUtil.createBusinessObject(lvBOName);
    }

    returnObj.setVerb(lvVerb);

    parseAttributeList(returnObj);

    return returnObj;
}
```

Figura 40. Esempio di metodo getBO() (Parte 1 di 4)

```
// Analizzare la Stringa e riepmire gli attributi nell'oggetto di business
protected void parseAttributeList(BusinessObjectInterface pvBO)
    throws Exception
{
    if ( mEndOfBOString )
        throw new Exception(...);
    else if( pvBO == null )
        throw new Exception(...);

    int lvAttrNum = pvBO.getAttrCount();

    String lvAttrName = null;
    String lvAttrValue = null;
    int lvAttrMaxLength = 0;

    try {
        for (int lvAttrIndex = 0; lvAttrIndex < lvAttrNum;
                lvAttrIndex++)
        {
            CxObjectAttr lvAttrSpec = pvBO.getAttrDesc(lvAttrIndex);
            lvAttrName = lvAttrSpec.getName();

            // Stabilire se l'attributo è un attributo semplice oppure un
            // contenitore di oggetti di business
            if (lvAttrSpec.isObjectType())
            {
                // Ottenere un token in base al BOCountSize
                lvAttrMaxLength = mBOCountSize;
                lvAttrValue =  this.getNextToken(mBOCountSize, true);
                String lvBOType = lvAttrSpec.getTypeName();
                Object lvAttrBOValue = null;

                if (lvAttrSpec.isMultipleCard())
                {
                    this.getMultipleCard(pvBO,lvAttrIndex,lvBOType,
                        lvAttrValue);
                }
                else {
                    this.getSingleCard(pvBO,lvAttrIndex,lvBOType,
                        lvAttrValue);
                }
            }
            else
            {
                // Ottenere il prossimo token in base al MaxLength dell'attributo
                lvAttrMaxLength = lvAttrSpec.getMaxLength();
                if (lvAttrMaxLength > 0)
                    lvAttrValue =  this.getNextToken(lvAttrMaxLength, false);
                else
                    lvAttrValue = null;
```

Figura 40. Esempio di metodo getBO() (Parte 2 di 4)

```
                // Per gli attributi di stringa semplici, impostati su null, impostare su
                // valori di CxIgnore o CxBlank configurati, oppure impostare sul
                // valore dell'attributo
                if (lvAttrValue == null )
                    pvBO.setAttrValue(lvAttrIndex, null);
                else if (lvAttrValue.equals(mCxIgnore)||
                      lvAttrValue.equals(""))
                    pvBO.setAttrValue(lvAttrIndex, null);
                else if (lvAttrValue.equals(mCxBlank)||
                      lvAttrValue.equals(" "))
                    pvBO.setAttrValue(lvAttrIndex, "");
                else
                    pvBO.setAttrValue(lvAttrIndex, lvAttrValue);
            }
        }
    }
}


// Compila un contenitore secondario con valori nella Stringa
protected void getMultipleCard(BusinessObjectInterface pvParentBO,
    int pvParentAttrIndex, String pvBOType, String pvObjectCountString)
    throws CW_BOFormatException, Exception
{
    try {
        if ( pvObjectCountString.equals(mCxIgnore) )
        {
            // trace message
        }
        else {
            int lvObjectCount = Integer.parseInt(pvObjectCountString);
            if ( lvObjectCount == 0)
            {
                // trace message with the number of objects in container
            }
            else if (lvObjectCount > 0)
            {
                // Esiste almeno un'istanza dell'oggetto nella stringa
                BusinessObjectInterface lvChildBO = null;

                // Per ogni istanza dell'oggetto secondario, analizzare la lista
                // degli attributi, poi aggiungere il contenitore di oggetti all'oggetto principal
                for (int lvObjectIndex = 0; lvObjectIndex < lvObjectCount;
                      lvObjectIndex++)
                {
                    lvChildBO = getBOFromString(pvBOType);
                    pvParentBO.setAttrValue(pvParentAttrIndex,lvChildBO);
                }
            }
        }
    }
}
```

Figura 40. Esempio di metodo getBO() (Parte 3 di 4)

```
// Compila un oggetto secondario a cardinaità singola con i valori nella Stringa
protected void  getSingleCard(BusinessObjectInterface pvParentBO,
    int pvParentAttrIndex, String pvBOType, String pvObjectCountString)
    throws CW_BOFormatException, Exception
{
    try {
        BusinessObjectInterface lvChildBO = null;

        // Verificare il token per il conteggio dell'oggetto
        // Se non corrispode ad "1", assumere che l'oggetto secondario debba
        //  essere null
        if (pvObjectCountString.equals("1"))
        {
            // La stringa contiene un'istanza singola dell'oggetto secondario
            lvChildBO = getBOFromString(pvBOType);
            pvParentBO.setAttrValue(pvParentAttrIndex, lvChildBO);
        }
        else if ( pvObjectCountString.equals(mCxIgnore) ||
              pvObjectCountString.equals("0"))
        {
            // Verificare che il token per il conteggio dell'oggetto sia valido
        }
        else
            throw new CW_BOFormatException(...);
    }
}
```

Figura 40. Esempio di metodo getBO() (Parte 4 di 4)

Implementazione della conversione da un oggetto di business

Il metodo astratto in Tabella 71 effettua la conversione da oggetto di business a stringa; ossia, ciascuno di essi crea dati serializzati in un particolare formato dall'oggetto di business.

Tabella 71. Metodi astratti per implementare la conversione da oggetto di business a stringa

Metodo astratto	Descrizione
getStringFromBO()	Converte i dati in un oggetto di business ad un oggetto `String` .
getStreamFromBO()	Converte i dati in un oggetto di business ad un oggett `InputStream` .
getByteArrayFromBO()	Converte i dati in un oggetto di business in un array di byte.

Lo scopo della conversione da un oggetto di business è di creare una forma serializzata di tutti i dati in un oggetto di business. A volte, però, alcuni dati in un oggetto di business *non* dovrebbero essere inclusi nei dati serializzati. Ad esempio, un oggetto di business potrebbe utilizzare un meta-oggetto secondario per contenere le informazioni di configurazione dinamica per un connettore.

IBM riserva tutte le informazioni specifiche sull'applicazione aventi il previsso `cw_mo_`*`label`* per la configurazione e/o meta-dati dinamici. Per indicare qualsiasi attributi che un gestore dati dovrebbe *ignorare* durante la conversione da oggetto di business, la definizione dell'oggetto di business per l'oggetto di business primario specifica la seguente tag nelle informazioni specifiche sull'applicazione:

`cw_mo_`*`label=nome_attributo_meta-oggetto_secondario`*

dove *`label`* è una stringa che viene definita per identificare ulteriormente lo scopo della tag `cw_mo_` e *`nome_attributo_meta-oggetto_secondario`* identifica il nome

dell'attributo da ignorare. Questo attributo di solito contiene il meta-oggetto secondario. Le tag cw_mo_*label* multiple sono delimitate da un punto e virgola (;).

Se vengono implementati i metodi getStringFromBO(), getStreamFromBO(), e getByteArrayFromBO() per un gestore dati personalizzato, i suddetti metodi devono assicurare che il gestore dati salti gli attributi specifici del connettore come segue:

1. Verificare l'esistenza di ogni tag cw_mo_*label* (dove *label* è una stringa fornita per identificare l'attributo da ignorare) nelle informazioni specifiche sull'applicazione della definizione dell'oggetto di business per l'iggetto di business che si sta convertendo.
2. Se esiste una tag cw_mo_*label* , localizzare la stringa fornita da questa tag (*nome_attributo_meta-oggetto_secondario*). Ignorare ogni spazio bianco intorno al segno di uguale(=).
3. Mentre si esegue un loop attraverso gli attributi dell'oggetto di business, confrontare ogni nome dell'oggetto con la stringa *nome_attributo_meta-oggetto_secondario* . Saltare ogni attributo con questo nome.

Il seguente frammento di codice mostra come saltare gli attributi:

```
List configObjList =
     com.crossworlds.DataHandlers.text.namevalue.listMOAttrNames(BusObj);

//questa lista contiene nomi di attributi che sono oggetti di configurazione
for ( attributes in BusObj )
{
          String attrName = BusObj.getAttrDisc(i).getName();
          if ( configObjList.contains(attrName) )
          {
            //saltare
            continuare;
          }
}
```

Ad esempio, si supponga che un oggetto di business chiamato MyCustomer utilizzi un meta-oggetto secondario per contenere informazioni di routing specifiche di un connettore. Se questo meta-oggetto è rappresentato da un attributo chiamato CustConfig MyCustomer potrebbe avere la seguente tag nelle informazioni specifiche sull'applicazione:

```
cw_mo_cfg=CustConfig
```

Durante la conversione da un oggetto di business, il gestore dati personalizzato verifica le informazioni sperifiche sull'applicazione per la definizione dell'oggetto di business associato a MyCustomer, localizza la tag cw_mo_cfg e stabilisce che l'attributo CustConfig deve essere saltato. I risultanti dati serializzati dal gestore dati *non* includono il meta-oggetto secondario CustConfig .

Nota: Durante la conversione da un oggetto di business, i gestori dati IBM saltano *tutti* gli attibuti identificati dalla tag cw_mo_*label* .

E' necessario sviluppare un gestore dati personalizzato per gestire le tag cw_mo_*label solo* se il gestore dati lavora con connettori che utilizzano meta-oggetti secondari o altri oggetti dinamici.

Implementazione del metodo getStringFromBO(): Il metodo getStringFromBO() effettua operazioni di conversione da oggetto di business a stringa; ossia, converte i dati contenuti in un oggetto di business in un oggetto String . Con questo metodo, il richiedente invia l'oggetto di business da convertire. Figura 41 mostra il

metodo getStringFromBO() quando implementato dal gestore dati FixedWidth. Il metodo crea una String di campi a larghezza fissa.

Questo esempio illustra i passaggi nella conversione di dati da un oggetto di business ad un oggetto Reader :

1. Il metodo getStringFromBO() richiama il metodo setAttrList() ed effettua ripetutamente un loop attraverso gli attributi nell'oggetto di business. Quando il metodo setAttrList() trova un attributo semplice richiama il metodo setSimpleToken() per impostare il valore.
2. Il metodo setSimpleToken() aggiunge il valore dell'attributo ad un oggetto StringBuffer , converte l'oggetto StringBuffer in un oggetto String ed aggiunge la stringa al metodo StringBuffer , che rappresenta l'intero oggetto di business. Dopo che il metodo setAttrList() ha elaborato l'oggetto di business, il metodo getStringFromBO() converte l'oggetto StringBuffer in un oggetto String e restutuisce l'oggetto String al richiedente.

```
String getStringFromBO(BusinessObjectInterface theObj pubblico,
    Object config)
    throws Exception
{
    traceWrite(
      "Entering getStringFromBO(BusinessObjectInterface, Object) "
      + " for object type " + theObj.getName(),
      JavaConnectorUtil.LEVEL4);

    clear(config);
    String lvBOString = null;
    setAttrList(theObj);
    lvBOString = mBOStringBuffer.toString();

    traceWrite(
      "Exiting getStringFromBO(BusinessObjectInterface, Object) "
       + " for object type " + theObj.getName(),
       JavaConnectorUtil.LEVEL4);

    return lvBOString;
}


protected void setAttrList(BusinessObjectInterface pvBO) throws Exception
{
    traceWrite(
      "Entering setAttrList(BusinessObjectInterface) for object "
      + pvBO.getName(), JavaConnectorUtil.LEVEL4);

    int lvAttrNum = pvBO.getAttrCount();

    String lvAttrName = null;
    String lvAttrValue = null;
    int lvAttrMaxLength = 0;

    // Aggiungere il nome ed il verbo dell'oggetto di business alla
    //  Stringa a larghezza fissa
    this.setSimpleToken( mBONameSize, pvBO.getName());
    this.setSimpleToken( mBOVerbSize, pvBO.getVerb());

    try {
        List moAttrNames = listMOAttrNames( pvBO );
        int lvAttrCount = pvBO.getAttrCount();

        ATTRIBUTE_WALK: for (int lvAttrIndex = 0;
              lvAttrIndex < lvAttrCount; ++lvAttrIndex)
        {
            CxObjectAttr lvAttrSpec = pvBO.getAttrDesc(lvAttrIndex);
            lvAttrName = lvAttrSpec.getName();

            // Verificare se l'attributo corrente sia di tipo semplice (Stringa) Check if the
            // oppure un oggetto contenuto.
            if (lvAttrSpec.isObjectType())
            {
                //saltare gli oggetti secondari designati come meta oggetti
                if( moAttrNames.contains( lvAttrName ) )
                {
                    continue ATTRIBUTE_WALK;
                }
```

Figura 41. Esempio di metodo getStringFromBO() (Parte 1 di 5)

```
if (lvAttrSpec.isMultipleCard())
{
    CxObjectContainerInterface lvAttrMultiCardBOValue =
  (CxObjectContainerInterface) pvBO.getAttrValue(lvAttrIndex);

    if (lvAttrMultiCardBOValue == null)
    {
        traceWrite(
    "setAttrList found empty multiple cardinality container "
    + lvAttrSpec.getTypeName(), JavaConnectorUtil.LEVEL5);

        // Aggiungere il conteggio alla Stringa a larghezza fissa
        this.setSimpleToken( mBOCountSize, "0");
    }
    else
    {
        int lvObjectCount =
          lvAttrMultiCardBOValue.getObjectCount();
        traceWrite(
      "setAttrList found multiple cardinality container "
      + lvAttrSpec.getTypeName() + " with "
      + lvObjectCount + " instances",
      JavaConnectorUtil.LEVEL5);

        // Aggiungere il conteggio alla Stringa a larghezza fissa
        this.setSimpleToken( mBOCountSize,
          Integer.toString(lvObjectCount));

        // Aggiungere ogni oggetto nel contenitore alla Stringa
        //  a larghezza fissa.
        for (int lvContObjectIndex = 0;
              lvContObjectIndex < lvObjectCount;
              ++lvContObjectIndex)
            setAttrList(
        lvAttrMultiCardBOValue.getBusinessObject(
            lvContObjectIndex));
    }
}
else
{
    BusinessObjectInterface lvAttrSingleCardBOValue =
  (BusinessObjectInterface) pvBO.getAttrValue(lvAttrIndex);

    if (lvAttrSingleCardBOValue == null)
    {
         traceWrite(
    "setAttrList found empty single cardinality container "
    + lvAttrSpec.getTypeName(), JavaConnectorUtil.LEVEL5);

         // Aggiungere il conteggio alla Stringa a larghezza fissa
         this.setSimpleToken( mBOCountSize, "0");
    }
```

Figura 41. Esempio di metodo getStringFromBO() (Parte 2 di 5)

```
                else
                {
                    traceWrite(
                  "setAttrList found single cardinality container "
                  + lvAttrSpec.getTypeName(),
                  JavaConnectorUtil.LEVEL5);

                    // Aggiungere il conteggio alla Stringa a larghezza fissa
                    this.setSimpleToken( mBOCountSize, "1");
                    setAttrList(lvAttrSingleCardBOValue);
                }
            }
        }
        else
        {
            lvAttrValue = (String) pvBO.getAttrValue(lvAttrIndex);
            lvAttrMaxLength = lvAttrSpec.getMaxLength();

            if (lvAttrMaxLength > 0)
                this.setSimpleToken(lvAttrMaxLength, lvAttrValue);
           }
        }
    }
    catch (CxObjectNoSuchAttributeException e)
    {
        throw new Exception(e.getMessage());
    }

    traceWrite(
        "Exiting setAttrList(BusinessObjectInterface) for object "
        + pvBO.getName(), JavaConnectorUtil.LEVEL4);
}


protected void setSimpleToken( int pvCellSize, String pvTokenValue)
    throws Exception
{
    traceWrite( "Entering setSimpleToken(int, String)",
      JavaConnectorUtil.LEVEL4);

    StringBuffer lvNewBuffer = new StringBuffer(pvCellSize);
    int lvTokenLength = 0;
    int lvCxIgnoreLength = mCxIgnore.length();
    int lvCxBlankLength = mCxBlank.length();

    int lvPadNumber = 0;


    // Verificare che il valore del token sia null
    if (pvTokenValue == null)
    {
        // In questo caso bisognerà aggiungere il valore configurato CxIgnore alla
        //  Stringa a larghezza fissa se corrisponde alla cella.
        if (!mTruncation && lvCxIgnoreLength > pvCellSize)
            throw new Exception(
                " Null attribute value encountered where cell size is "
                + pvCellSize + ", size of CxIgnore value is "
                + lvCxIgnoreLength + "and trucation is not allowed. "
                + "Please check your MO format configuration.");
```

Figura 41. Esempio di metodo getStringFromBO() (Parte 3 di 5)

```
        else
        {
            lvPadNumber = pvCellSize - lvCxIgnoreLength;
            lvNewBuffer.append(mCxIgnore);
        }

    }
    else if (pvTokenValue.equals(""))
    {
        // In queto caso, il valore configurato CxBlank viene aggiunto
        //  alla Stringa a larghezza fissa se corrisponde alla cella.
        if (! mTruncation && lvCxBlankLength > pvCellSize)
            throw new Exception(
              " Blank attribute value encountered where cell size is "
              + pvCellSize + ", size of CxBlank value is "
              + lvCxBlankLength + "and trucation is not allowed. "
              + "Please check your MO format configuration.");
        else
        {
            lvPadNumber = pvCellSize - lvCxBlankLength;
            lvNewBuffer.append(mCxBlank);
        }
    }
    else
    {
        // In questo caso, aggiungere il valore del token alla
        // Stringa a larghezza fissa, a meno che i dati siano troppo lunghi rispetto alla cella.
        lvTokenLength = pvTokenValue.length();
        if (!mTruncation && lvTokenLength > pvCellSize )
            throw new Exception(
              " Attribute value encountered where cell size is "
              + pvCellSize + ", size of token value is "
              + lvTokenLength + "and trucation is not allowed. "
              + "Please check your MO format configuration.");
        else
        {
            lvNewBuffer.append(pvTokenValue);
            lvPadNumber = pvCellSize - lvTokenLength;
        }
    }

    if (lvPadNumber <= 0 && mTruncation)
        // Il token è più lungo della cella ed è permesso il troncamento,
        //  così che i caratteri fino a pvCellSize vengano aggiunti
        lvNewBuffer.setLength(pvCellSize);
    else if (lvPadNumber > 0)
    {
        // Allineare la cella in base all'opzione di configurazione scelta
        if ( mAlignment.equals(fixedwidth.AlignmentLeft) ||
              mAlignment.equals(fixedwidth.AlignmentBoth))
            this.padRight(lvNewBuffer, lvPadNumber);
        else if (mAlignment.equals(fixedwidth.AlignmentRight))
            this.padLeft(lvNewBuffer, lvPadNumber);
    }
```

Figura 41. Esempio di metodo getStringFromBO() (Parte 4 di 5)

```
    String lvNewBuffString =  lvNewBuffer.toString();

    // E' bene notare che in questo caso è possibile che si verifichino ptoblemi legati alle
    //  quando il livello di traccia è basso, ma nella maggior parte dei casi questo non succede p
    //  non è *di solito* molto lungo.
    traceWrite(
      "Adding the following token to the fixed width String: "
      + lvNewBuffString, JavaConnectorUtil.LEVEL5);

    // Dopo che la cella è stata completamente formattata, aggiungere alla Stringa a
    //  larghezza fissa che si sta creando.
    mBOStringBuffer.append(lvNewBuffString);

    traceWrite( "Exiting setSimpleToken(int, String)",
      JavaConnectorUtil.LEVEL4);
}
```

Figura 41. Esempio di metodo getStringFromBO() (Parte 5 di 5)

Implementazione del metodo getStreamFromBO(): Il metodo `getStreamFromBO()` converte i dati in un oggetto di business in un oggetto `InputStream` . Figura 42 mostra un esempio di implementazione del metodo `getStreamFromBO()` . In questa implementazione, il metodo `getStreamFromBO()` richiama `getStringFromBO()` per creare un oggetto `String` contenente i dati di un oggetto di business e converte l'oggetto `String` in un oggetto `InputStream`. Il metodo restituisce un oggetto `InputStream` che rappresenta i dati nell'oggetto di business.

```
InputStream getStreamFromBO(BusinessObjectInterface theObj pubblico,
    Object config)
    throws Exception
{
    clear(config);

    String BOstring;
    BOstring = getStringFromBO(theObj, config);

    return new ByteArrayInputStream( BOstring.getBytes() );
}
```

Figura 42. Esempio di metodo getStreamFromBO()

Sovrascrizione di metodi pubblici

Oltre ai metodi astratti `DataHandler` (che vanno *necessariamente* implementati), bisogna anche sovrascrivere alcuni metodi pubblici della classe `DataHandler` (consultare Tabella 72) in modo che essi possano lavorare in maniera ottimale con il gestore dati personalizzato.

Tabella 72. Metodi pubblici della classe DataHandler

Metodo pubblico DataHandler	Descrizione
getBO() - public	Converte dati serializzati (in uno qualsiasi dei molteplici formati) in un oggetto di business.
getBOName()	Ottiene il nome dell'oggetto di business dai dati serializzati.
getBooleanOption()	Ottiene il valore di un'opzione di configurazione booleana dal gestore dati.
getOption()	Ottiene il valore di un'opzione di configurazione dal gestore dati.
setOption()	Imposta un'opzione di configurazione nel gestore dati.
traceWrite()	Richiama un metodo trace-write per il contesto apporpriato del gestore dati: connettore o il Server Access Interface.

Creazione di un gestore nome personalizzato

Un gestore dati può richiamare un *gestore NNnome* per estrarre il nome della definizione dell'oggetto di business dai dati serializzati. Questa operazione è necessaria durante la conversione da stringa ad oggetto di business, quando il richiedente del gestore dati personalizzato*non* passa un oggetto di business che deve essere compilato con dati serializzati. In questo caso, il gestore dati deve creare l'oggetto di business prima di poterlo compilare. Prima di creare l'oggetto di business, il gestore dati deve sapere il nome della definizione dell'oggetto di business ad esso associato. Ed è il gestore nome che ottiene questo nome dell'oggetto di business.

Nota: Attualmente, i gestori dati XML ed EDI utilizzano gestori nome per ottenere il nome dell'oggetto di business da creare.

L'attività di creazione di un gestore nome personalizzato include i seguenti passaggi generali:

1. Creare una classe che estenda la classe `NameHandler` .
2. Implementare il metodo astratto `getBOName()` che legga i dati serializzati e restituitsca il nome dell'oggetto di business.
3. Compilare la classe ed aggiungerla al file `DataHandlers\CustDataHandler.jar` . Per ulteriori informazioni, consultare "Inserimento di un gestore dati nel file jar" a pagina 195.
4. Impostare il valore di default dell'attributo `NameHandlerClass` nel meta-oggetto per il gestore dati.

Estensione della classe NameHandler

Per creare un gestore dati personalizzato, estendere la classe base del gestore nome(`NameHandler`) in modo da poter creare la propria *classe del gestore-nome*. La classe `NameHandler` contiene il metodo per estrarre il nome di un oggetto di business dai dati serializzati. Il pacchetto per questa classe base del gestore-nome è `com.crossworlds.DataHandlers.NameHandler`.

Per dividere la classe name-handler, seguire questi passaggi:

1. Creare una classe del gestore nome che estenda la classe NameHandler.
2. Assocurarsi che il file di classe del gestore nome importi le classi del pacchetto NameHandler

   `importare`
3. Implementare il metodo `getBOName()` , ossia il metodo astratto nella classe `NameHandler` . Per ulteriori informazioni, consultare "Implementazione del metodo getBOName()" a pagina 195.

La definizione della classe `NameHandler` avviene come segue:

```
// Importa
importare java.lang.String;
importare java.io.Reader;
importare com.crossworlds.DataHandlers.Exceptions.MalformedDataException;

classe astratta pubblica NameHandler {

// Costruttori
NameHandler() pubblico { }
```

```
// Metodi
public abstract String getBOName(Reader serializedData,
String boPrefix)
throws MalformedDataException;
}

/* Questo metodo è stato introddotto affinche il  NameHandler avesse un
 * riferimento nel DataHandler. La base DataHandler richiama questo
 * metodo dopo aver istanziato il NameHandler:
 * es. nh = (NameHandler)Class.forName(className).newInstance();
 *     nh.setDataHandler(this);
*/
public final void setDataHandler( DataHandler dataHandler )
{
dh = dataHandler;
}
```

Per creare il proprio gestore nome, estendere la classe base astratta NameHandler .

Implementazione del metodo getBOName()

L'estensione dellla classe NameHandlerrichiede l'implementazione del metodo getBOName() , che legge i dati derializzati e restituisce il nome dell'oggetto di business associato a quei dati. La sintassi per questo metodo è:

```
public abstract String getBOName(Reader serializedData, String BOPrefix)
invia MalformedDataException
```

dove:

serializedData
: é il riferimento ad un oggetto che contiene il messaggio.

BOPrefix
: E' un valore String che contiene il prefisso dell'oggetto di business per i nomi delle definizioni degli oggetti di business; questo argomento può essere impostato al attributo BOPrefixnel meta-oggetto (se esistente).

Impostazione dell'attributo del meta-oggetto

Per comunicare al gestore dati che deve utilizzare un gestore nome personalizzato, occorre impostare la propietà del valore predefinito dell'attributo di un meta-oggetto sul nome classe completo. Il gestore dati può così ottenere il nome classe da una delle opzioni di configurazione durante l'esecuzione. Per definizione, questo attributo del meta-oggetto viene chiamatoNameHandlerClass. Il meta-oggetto secondario associato ad entrambi i gestori dati XML ed EDI include questo attributo. Il valore prededefinito IBM di questo attributo specifica il nome della classe predefinita del gestore nome. Per ottenere un gestore dati utilizzare un gestore nome personalizzato, assicurarsi che la proprietà del Valore Predefinito per l'attributo NameHandlerClass nel meta-oggetto secondario associato al gestore dati che si vuole estendere venga aggiornata.

Inserimento di un gestore dati nel file jar

Dopo aver completato il codice per il nuovo gestore dati, bisogna compilare la classe ed aggiungerla al file archivio (jar) JAVA. Il file CustDataHandler.jar contiene i gestori dati personalizzati. Tale file jar è risiede nella sottodirectory DataHandlers della direcotry del prodotto. Per localizzare una classe del gestore dati, il metodo createHandler() cerca questo file jar dopo aver cercato il file CwDataHandler.jar che contiene i gestori dati forniti.

Nota: Per poter compilare un codice Java, bisogna aver istallato il Java Development Kit (JDK) sulla propria macchina. Per informazioni sulla versione del JDK richiesto e istruzioni per la sua installazione consultare la guida all'installazione del prod otto.

Per aggiungere un gestore dati personalizzato al `CustDataHandler.jar`:

1. Modificare lo lo script di compilazione del gestore dati per aggiungere i nomi dei file sorgente Java.

   Lo script di compilazione del gestore dati risiede nella seguente sottodirectory della directory del prodotto:

   ```
   DevelopmentKits\edk\DataHandler
   ```

   Windows

   In un sistema lo script di compilazione di un gestore dati viene chiamato `make_datahandler.bat`. Per aggiungere i nomi dei file sorgente Java alla riga:

   ```
   impostare SOURCE_FILES_DH=
   ```

   Linux

   In un sistema Linux, lo script di compilazione del gestore dati viene chiamato: `make_datahandler`. Per aggiungere i nomi dei file sorgente Java alla riga:

   ```
   SOURCE_FILES_DH=
   ```

2. Eseguire lo script di compilazione del gestore dati per compilare i file Java in un file `.class` .
3. Aggiungere la nuova classe al file `CustDataHandler.jar` utilizzando i seguenti comandi:

   ```
   jar -vf CustDataHandler.jar file_input
   ```

   dove *file_input* è una lista di file di classe da aggiungere.

Creazione dei meta-oggetti di un gestore dati

Se si scrive un gestore dati personalizzato che utlizza meta-oggetti del gestore dati, bisogna:

- Creare un meta-oggetto del gestore dati secondario che contiene gli attributi delle informazioni di configurazione del gestore dati personalizzato.
- Modificare il meta-oggetto del gestore dati di livello superiore in modo da includere il tipo MIME affinchè il gestore dati possa essere configurato quando viene istanziato.
- Impostare il meta-oggetto nel sistema di integrazione business.

Nota: Per un supporto nella scelta di utilizzare o meno meta-oggetti nella progettazione di un gestore dati, consultare"Utilizzo dei meta-oggetti del gestore dati" a pagina 179.

Creazione del meta-oggetto secondario

Il meta oggetto secondario contiene informazioni di configurazione per il gestore dati. Il metodo `createHandler()` utilizza queste informazioni per inizializzare il

gestore dati appena istanziato. Per informazioni più dettagliate su questo processo, consultare"Utilizzo di un tipo MIME" a pagina 14.

Per creare un meta-oggetto secondario per un gestore dati personalizzato:

1. Creare un meta-oggetto secondario per rappresentare un'istanza del gestore dati.

   Per creare tale meta-oggetto si può utilizzare il tool di progettazione oggetti di business Business Object Designer Express. Il meta-oggetto deve contenere attributi per definire le informazioni do configurazione richiesti dal gestore dati. Il meta-oggetto secondario *deve* almeno avere una `attributo ClassName` .
2. Stabilire se occorre specificare il nome della classe del gestore dati nell'attributo `ClassName` .
   - Per un gestore dati in formato predefinito non occorre specificare un valore per il`ClassName`. Il formato predefinito è:

     `com.crossworlds.DataHandlers.`*`MimeTypeString`*

     In ogni caso, è possibile specificare il nome della classe per il gestore dati nel valore predefinito dell'attributo `ClassName` .
   - Per una classe di gestore dati che non sia in formato predefinito occorre specificare il nome di classe completo per l'istanza del gestore dati. Altrimenti il metodo `createHandler()` non riesce a localizzare la classe del gestore dati quando prova ad istanziare il gestore dati.
3. Impostare i valori predefiniti negli attributi rilevanti nel meta-oggetto secondario per configurare la modalità in cui l'istanza del gestore dati elaborerà i dati.

Modifica del meta-oggeto di livello superiore

Quando un richiedente fornisce un tipo MIME al `createHandler()` , il `createHandler()` stabilisce quale gestore dati istanziare nel modo seguente:

1. Localizzare il nome del meta-oggetto di livello superiore associato al gestore dati
2. Cercare in questo meta-oggetto di livello superiore il tipo MIME che corrisponda ai dati da convertire.
3. Se questo tipo MIME esiste, trovare il nome del meta-oggetto secondario ad esso associato contenente le informazioni di configurazione.

Per una spiegazione più dettagliata di questo processo, consultare"Utilizzo di un tipo MIME" a pagina 14. Perchè questo processo venga completato correttamente, il `createHandler()` deve riuscire a localizzare il tipo MIME associato a quei dati. Per questo è necessario modificare il meta-oggetto di livello superiore del gestore dati in modo da includere un attributo per il tipo MIME dei dati che il gestore dati converte. Questo attributo deve includere:

- Un nome che corrisponda alla stringa di tipo-MIME per il tipo MIME associato del gestore dati.

  Il nome del Il tipo MIME può contenere *solo* caratteri alfanumerici e di sottolineatura (_). Nessun altro carattere speciale è valido. Se il tipo MIME contiene un puntoIf your MIME (.)bisognerà sosttuirlo con un carattere di sottolineatura.
- Un oggetto di business a cardinalità singola
- Un tipo che sia il nome del meta-oggetto secondario che rappresenta il gestore dati

Ad esempio, Figura 43 mostra il meta-oggetto del connettore di livello superiore per un gestore dati HTME personalizzato.

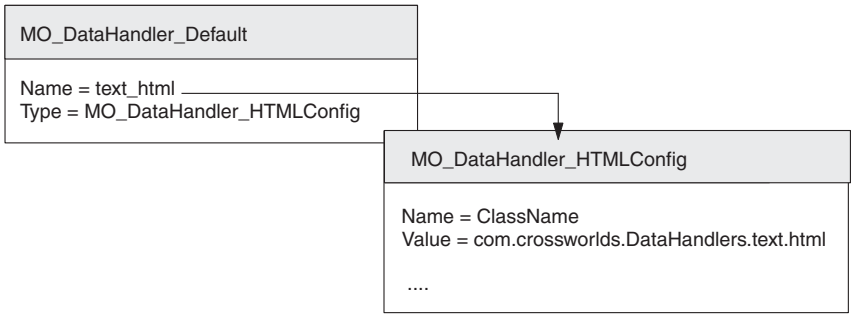

Figura 43. Esempio di meta-oggetto del connettore di livello superiore del gestore dati persoalizzato

In Figura 43, il meta-oggetto di livello superiore predefinito per un connettore `MO_DataHandler_Default`) è stato modificato per supportare un nuovo tipo MIME: HTML. A supporto di questo tipo MIME, il meta-oggetto `MO_DataHandler_Default` contiene le seguenti proprietà dell'attributo:

```
Nome Attributo = text_html
Tipo Attributo = MO_DataHandler_HTMLConfig
```

Importante: Il nome del tipo MIME può essere composto solo da caratteri alfanumerici e di sottolineatura (_). Nessun altro carattere speciale può essere usato per questo tipo MIME.

Impostazione del meta-oggetto nel sistema di integrazione business

Dopo aver create il meta-oggetto del gestore-dati è necessario impostare questi meta-oggetti nel sistema WebSphere Business Integration Server Express come segue:

1. Ciaricare i nuovi meta-oggetti nel repository.
2. Modificare l'apposito meta-oggetto a seconda del contesto in cui il gestore dati verrà richiamato:
   - Se il gestore dati dovrà essere eseguito nel contesto di un connettore, aggiungere il meta-oggetto del gestore dati come secondario al meta-oggetto di livello superiore del gestore dati. Successivamente aggiungere il supporto per il meta-oggetto di livelo superiore del gestore dati alla definizione del connettore.
   - Se si sta utilizzando InterChange Server Express e il gestore dati dovrà essere utilizzato nel contesto del Server Access Interface, aggiungere livello superiore del server `MO_Server_Datahandler`.

Impostazione dei altri oggetti di business

Oltre a codificare il gestore dati, è necessario impostare tutti gli oggetti di business per il gestore dati. Creare oggetti di business conformi ai requisiti del gestore dati e a quelli del richiedente.

Suggerimenti: Assicurarsi che tutti gli oggetti di business richiesti dal gestore dati siano presenti nellalista degli oggetti supportati di tutti i connettori utilizzati dal gestore dati.

Configurazione di un connettore

Se un gestore dati verrà utilizzato nel contesto di un connettore, bisognerà configurare ogni connettore in modo da ottenere il nome del connettore di livello superiore meta-oggetti. I connettori ottengono le informazioni sui nomi e nomi classe dei meta-oggetti del gestore dati in diversi modi. Ad esempio:

- Per configurare Adapter affinché XML utilizzi un gestore dati, occorre consigurare la proprietà di configurazione del connettore `DataHandlerConfigMO` e l'attributo `MimeType` nell'oggetto di business del connettore XML.
- Per configurare Adapter in modo tale che JText utilizzi un gestore dati, occorre impostare `ClassName` o `DataHandlerConfigMO` e gli attributi `MimeType` nel meta-oggetto di configurazione JText.

Per ulteriori informazioni, consultare "Configurazione dei connettori per l'utilizzo dei gestori dati" a pagina 26.

Suggerimenti: Durante la configurazione di un connettore che utilizzi un gestore dati, assicurarsi che l'ortografia del nome del meta-oggetto e del tipo MIME sia corretta.

Un gestore dati internazionalizzato

Un *gestore dati internazionalizzato* è un gestore dati scritto in modo tale che possa essere personalizzato per una particolare locale. Una *locale* è una parte dell'ambiente di un utente che porta contemporaneamente informazioni su come trattare i dati specifici sul territorio, la lingua e il paese dell'utente finale. La locale è solitamenteinstallata come parte del sistema operativo. La creazione di un gestore dati che tratti i dati sensibili di una locale viene chiamata *internazionalizzazione* (I18N) del gestore dati. La preparazione di un gestore dati internazionalizzato per una particolare locale viene chiamata *localizzazione* (L10N) del gestore dati.

Questa sezione fornisce le seguenti informazioni su un gestore dati internazionalizzato:

- Cos'è una locale?
- Considerazioni per la progettazione di un gestore dati internazionalizzato

Cos'è una locale?

Una *locale* fornisce le seguenti informazioni sull'ambiente dell'utente:

- Convenzioni culturali secondo la lingua ed il paese (o territorio)
  - Formati di dati:
    - Date: definire nomi completi ed abbreviati per i giorni della settimana ed i mesi e la struttura della data (compreso il separatore di data).
    - Numeri: definire i simboli per il separatore di migliaia ed i punti decimali e dove tali simboli sono ubicati all'interno del numero.
    - Orari: definire gli indicatori per gli orari espressi nelle 12 ore (come gli indicatori AM e PM) e la struttura dell'orario.
    - Valori monetari: definire i simboli numerici e di valuta e dove tali simboli sono ubicati all'interno del valore monetario.
  - Ordine di fascicolazione: come ordinare i dati per la serie di codici di caratteri particolari e della lingua.
  - La gestione della Stringa include compiti come il confronto di lettere (maiuscolo e minuscolo), sottostringhe e concatenazione.

- La *codifica carattere* —l'associazione da un carattere(una lettera dell'alfabeto) ad un valore numerico in una serie do codici di caratteri. Ad esempio, la serie di codici di caratteri ASCII codifica la lettera "A" come 65, mentre la serie di caratteri EBCIDIC codifica questa lettera come 43. La *serie di codici di caratteri* contiene docifiche per tutti i caratteri in uno o più alfabeti linguistici.

Un nome locale dispone del seguente formato:
ll_TT.codeset

dove *ll* è un codice di lingua a due caratteri (di solito in minuscolo), *TT* è un codice di territorio e paese a due lettere(di solito in maiuscolo) e *codeset* è il nome della serie di codici di caratteri associati. La parte *codeset* del nome è spesso facoltativa. La locale viene di solito installata come parte dell'installazione del sistema operativo.

Considerazioni per la progettazione di un gestore dati internazionalizzato

Questa sezione fornisce le seguenti categorie di considerazioni di progettazione pr l'internazionalizzazione di un gestore dati:

- Principi di progettazione sensibili alla locale
- Principi di progettazione della codifica carattere

Principi di progettazione sensibili alla locale

Per essere internazionalizzato, un gestore dati deve essere codificato come sensibile alla locale, ossia: il suo comportamento deve tenere in considerazione le impostazioni della locale ed eseguire il compito adatto a quella locale. Ad esempio, per locali che usano la lingua inglese il gestore dati deve ottenere i messaggi di errore dal file dei messaggi in lingua inglese. La struttura del gestore dati installata con il prodotto è internazionalizzata. Per completare l'internazionalizzazione (I18N) di un gestore dati che si intende sviluppare bisogna assicurarsi che l'implementazione del gestore dati sia internazionalizzata.

Un gestore dati internazionalizzato deve essere conforme ad una serie di proincipi di progettazione sensibili alla locale:

- I testi dei messaggi di errore, stato e traccia dovrebbero essere isolati dall'implementazione del gestore dati in un file di messaggi e traadotti nella lingua della locale.
- Le funzioni di fascicolazione ed ordinamento dei dati utilizzano un ordine di fascicolazione appropriato alla lingua ed al paese della locale.
- L'elaborazione di una stringa (ad esempio confronto, sottostringhe e caratteri maiuscolo/minuscolo) deve essere conforme ai caratteri nella lingua della locale.
- Il formato di date, numeri ed orari devono essere conformi alla locale.

Il gestore dati potrebbe aver bisogno di eseguire una elaborazione sensibile alla locale (ad esempio conversioni del formato dei dati) durante una conversione da applicazione di dati serializzati ad oggetto di business. Per tracciare la locale associata al ambiente del gestore dati, la classe `DataHandler` ha una variabile locale privata che viene inizializzata alla locale del sistema operativo sul quale il gestore dati è in esecuzione. E' possibile accede alla locale dell'ambiente del gestore dati (il valore di questa variabile `locale` privata) durante l'esecuzione attraverso i metodi accessori in Tabella 73.

Tabella 73. Metodi di accesso alla locale dell'ambiente del gestore dati

Classe Gestore Dati	Metodo
`DataHandler`	getLocale(), setLocale()

Quando viene creato un oggetto di buisiness viene associata una locale ai suoi dati. Questa locale si applica ai dati nell'oggetto di business, *non* al nome della definizione dell'oggetto di business o ai suoi attributi (che devono essere caratteri appartenenti alla serie di codici associati alla locale in inglese americano) `en_US`). Per creare un oggetto di business il gestore dati può utilizzare i metodi illustrati in Tabella 74. Questi metodi hanno accesso alla variabile locale privata nella classe `DataHandler` . Quando uno di questi metodi crea un oggetto di business associa a tale oggetto di business la locale che la variabile locale privata `DataHandler` specifica.

Nota: Questi metodi in Tabella 74 impostano la locale *solo* nell'oggetto di business di livello superiore. Se l'oggetto di business contiene oggetti di business secondari, tali oggetti secondari potrebbero non avere una locale valida poiché essi hanno la loro locale impostata sul sistema predefinito.

Usare questi medoti in Tabella 74 per creare un oggetto di business ed impostare la locale per i suoi dati. Per assicurars che la variabile locale privata sperifichi la locale corretta per i dati nell'oggetto di business è possibile utilizzare il metodo `setLocale()` *prima* di richiamare qualsiasi metodo in Tabella 74.

Tabella 74. Metodi di assegnazione di una locale ad un oggetto di business

Classe Gestore Dati	Metodo
`DataHandler`	getBO() - public, getBOName()

Principi di progettazione della codifica carattere

Se i dati vengono trasferiti da una posizione che utilizza una serie di codici ad una posizione che utilizza una serie di codici differenti, occorrerà eseguire una conversione dei caratteri per i dati che mantengono il loro significato. L'ambientedi esecuzione Java all'interno del Java Virtual Machine (JVM) rappresenta i dati nel set di caratteri Unicode. Il set di caratteri Unicode è un set di caratteri universale che contiene codifiche per caratteri presenti nella maggior parte delle serie di codici di caratteri conosciute (sia a byte singolo che multipli). Esistono diversi formati di codifica nel set Unicode. Le seguenti codifiche sono quelle più frequentemente utilizzate all'interno di sistemi di integrazione business:

- Ottetto multiplo universale di set di caratteri codificati: UCS-2

  Il set di codifica UCS-2 è il seto di caratteri Unicode codificato in 2 byte (ottetti)
- UCS Transformation Format, versione a 8-bit: UTF-8

  Il set di codifica UTF-8 è progettato per indirizzare l'uso dei dati dei caratteri Unicode negli ambienti Linux . Supporta i valori del codice ASCII (0...127) in modo che essi vengano sempre inequivocabilmente interpretati come un vero codice ASCII. Ogni valore di codice di solito è rappresentato come un valore 1-, 2-, o 3- byte.

La maggior parte dei componenti nel sistema IBM WebSphere Business Integration Server Express sono scritti in Perciò quando i dati vengono trasferiti all'interno della maggior parte dei componenti del sistema essi vengono codificati secondo il set di caratteri UNicode e non occorre effettuare alcuna conversione.

Poichè un gestore dati è un componente scritto in Java, esso tratta i dati serializzati nel set di codici Unicode. Di solito, la fonte di questo flusso di immissione dei dati viene anch'essa elaborata in Unicode. Perciò, di solito un gestore dati non deve eseguire alcuna conversione di carattere sui dati serializzati. Comunque, se i dati di ingresso e di uscita contengono un array di byte la cui codifica carattere non sia la stessa del sistema predefinito, il gestore dati deve fornire la codifica carattere.

Per tracciare la codifca carattere asociata all'ambiete del gestore dati, la classe `DataHandler` ha una variabile privata di codifica carattere inizializzata alla codifica carattere associata alla locale del sisytema operativo su cui il gestore dati è inn esecuzione. E' possibile accedere alla codifica carattere dell'ambiente del gestore dati (il valore di tale variabile privata do codifica carattere) durante l'esecuzione attraverso i metodi accessori in Tabella 75..

Tabella 75. Metodi di recupero della codifica carattere del gestore dati

Classe Gestore Dati	**Metodo**
`DataHandler`	getEncoding(), setEncoding()

Capitolo 11. Metodi della classe base del gestore dati

La classe `DataHandler` è la classe base per i gestori dati. E' contenuta nel pacchetto `com.crossworlds.DataHandlers`. Tutti i gestori dati, inclusi quelli personalizzati, devono estendere tale classe astratta.

I metodi illustrati in questo capitolo riguardano tre categorie:

- Metodi astratti che devono essere implementati
- Metodi pubblici che hanno un'implementazione fornita che, se necessario, dovrebbe essere sovrascritta
- Metodi statici che vengono richiesti dai connettori o dai client di accesso

Tabella 76 elenca i metodi della classe `DataHandler`.

Tabella 76. Metodi membri della classe DataHandler

Metodi membri	Tipo	Descrizione	Pagina
createHandler()	Pubblico Statico	Creazione di un'istanza di un gestore dati.	203
getBO() - abstract	Astratto	Compila un oggetto di business con i valori estratti da dati di immissione serializzati.	205
getBO() - public	Pubblico	Converte i dati serializzati in un oggetto di business.	206
getBOName()	Pubblico	Ottiene il nome dell'oggetto di business in base al contenuto dei dati serializzati.	207
getBooleanOption()	Pubblico	Ottiene il valore dell'opzione di configurazione specificata del gestore dati se contiene dati booleani.	209
getByteArrayFromBO()	Pubblico	Serializza un oggetto di business in un array di byte.	209
getEncoding()	Pubblico	Recupera la codifica carattere che il gestore dati utilizza.	210
getLocale()	Pubblico	Recupera la locale del gestore dati.	210
getOption()	Pubblico	Ottiene il valore dell'opzione di configurazione specificata del gestore dati (se è stata impostata).	211
getStreamFromBO()	Astratto	Serializza un oggetto di business in un oggetto InputStream.	212
getStringFromBO()	Astratto	Serializza un oggetto di business in un oggetto stringa.	212
setConfigMOName()	Statica	Imposta il nome del meta oggetto del gestore dati di livello superiore in una proprietà statica della classe base DataHandler.	213
setEncoding()	Pubblico	Imposta la codifica carattere che il gestore dati utilizza.	214
setLocale()	Pubblico	Imposta la locale del gestore dati.	214
setOption()	Pubblico	Imposta il valore dell'opzione di configurazione specificata del gestore dati.	215
traceWrite()	Pubblico	Richiama la funzione trace-write appropriata per scrivere un messaggio di traccia per il gestore dati.	216

createHandler()

Creazione di un'istanza di un gestore dati.

Syntax

```
public static DataHandler createHandler(String className,
    String mimeType, String BOPrefix);
```

Parametri

className E' il nome della classe dell'istanza del gestore dati da creare. Se non viene specificata, il metodo utilizza l'argomento *mimeType* per determinare su quale classe del gestore dati deve avviare un'istanza.

mimeType Specifica il tipo MIME dell'istanza del gestore dati da creare. Se non viene fornito, il metodo presume che un valore *className* deve essere fornito. Chiave al meta oggetto. Se *BOPrefix* viene fornito, *mimeType* diviene parte della chiave.

BOPrefix E' un parametro facoltativo. Se presente, viene combinato con *mimeType* per formare la chiave al meta oggetto. Questo argomento può essere utilizzato per specificare un sottotipo MIME. Inoltre, può essere utilizzato per impostare la proprietà di configurazione del gestore dati `BOPrefix`.

Valori di restituzione

Un'istanza di un gestore dati.

Eccezioni

Eccezione
L'eccezione viene emessa se il metodo non è in grado di creare un'istanza per il gestore dati.

Note

Questo metodo crea un'istanza per il gestore dati in base ai valori dei relativi parametri *className*, *mimeType* e *BOPrefix*:

- Se il metodo `createHandler()` viene richiesto da un connettore, il connettore può specificare un valore *className*. Se *className* viene specificato, `createHandler()` crea un'istanza per il gestore dati di questa classe del nome.
- Se *mimeType* viene specificato, `createHandler()` crea un gestore dati in base al tipo MIME specificato.

  Il metodo controlla il meta oggetto del gestore dati di livello superiore per un attributo il cui nome corrisponda al tipo contenuto o del parametro *mimeType* o alla combinazione *mimeType* e *BOPrefix*. Se viene trovato un attributo corrispondente, il valore dell'attributo `ClassName` nel meta oggetto secondario è utilizzato come nome della classe.

Se il metodo riesce nell'avvio di un'istanza per una classe del gestore dati, richiede `setupOptions()` per avviare le proprietà di configurazione per l'utilizzo da parte dell'istanza del gestore dati. Per una descrizione completa su come creare un'istanza `createHandler()` per un gestore dati, fare riferimento a "Istanziare il gestore dati" a pagina 12.

Ad esempio, per `MIME = "text/xml-application-xxx"`, il metodo carica la classe `com.crossworlds.DataHandlers.text.xml_application_xxx`.

getBO() - abstract

Compila un con un oggetto di business con valori estratti da dati di immissione serializzati.

Syntax

```
public abstract void getBO(Reader serializedData,
        BusinessObjectInterface theBusObj, Object config);
public abstract BusinessObjectInterface getBO(Reader serializedData,
        Object config);
```

Parametri

serializedData	E' un oggetto Java `Reader` che accede ai dati serializzati.
theBusObj	E' l'oggetto di business da compilare con i dati.
config	E' un oggetto facoltativo che contiene informazioni aggiuntive di configurazione per il gestore dati.

Valori di restituzione

Il primo modulo di questo metodo non ha valori di restituzione. Il secondo modulo restituisce un oggetto di business.

Note

Questo `getBO()` metodo è il metodo astratto che effettua l'operazione di conversione stringa ad oggetto di business per un gestore dati; ossia, definisce il modo di convertire dati serializzati generici (a cui si accede con un oggetto `Reader`) in un oggetto di business. Questo metodo ha due moduli:

- Il primo modulo compila gli oggetti di business vuoti, *theBusObj*, che sono passati dal richiedente.
- Il secondo modulo crea un'istanza dell'oggetto di business e la compila.

Nota: Un gestore dati che è richiesto nel contesto di Server Access Interface richiede di fornire funzionalità *solo* per il secondo modulo del metodo `getBO()`.

Importante: Il metodo `getBO()` è un metodo astratto che non ha implementazioni predefinite.Perciò, la classe del gestore dati *deve* implementare questo metodo.

E' necessario che i dati serializzati siano passati in `getBO()` così come in un oggetto Java `Reader`. Tuttavia, poiché `Reader` è una classe base, è possibile realmente passare un'istanza di una delle numerose sottoclassi della classe `Reader`. Alcune delle sottoclassi `Reader` forniscono un'implementazione per l'operazione `mark()` ed altre no. L'operazione `mark()` consente al richiedente di contrassegnare una particolare posizione all'interno di un flusso e quindi di ritornare successivamente a quella posizione.

Nota: Per passare un oggetto `Reader` nel metodo `getBO()` dei XML o gestori dati EDI, è necessario assicurarsi che la sottoclasse `Reader` implementi il metodo `mark()`. E' possibile richiedere il metodo `isMarkSupported()` della classe `Reader` per determinare se tale metodo è supportato dall'oggetto `Reader` che si sta utilizzando. Si raccomanda di passare nei dati serializzati come un oggetto `StringReader`.

Se è necessario fornire al proprio gestore dati maggiori informazioni di configurazione di quelle incluse nel meta oggetto, è possibile utilizzare l'opzione *config* per passare ad un oggetto che contiene queste informazioni. Ad esempio, *config* potrebbe essere un file maschera o una stringa ad un URL per uno schema che viene utilizzato per creare un documento XML da un oggetto di business.

Se *config* è un tipo di oggetto di business, è possibile implementare il metodo `getBO()` per richiedere `setupOptions(`*config*`)`. Il metodo `setupOptions()` è definito nella classe base `DataHandler`. Tale metodo utilizza i nomi dell'attributo nell'oggetto di business come nomi della proprietà e i valori predefiniti come valori per quelle proprietà. Imposta i valori delle proprietà di configurazione nell'oggetto per l'utilizzo da parte del gestore dati.

Una volta implementato il metodo astratto `getBO()`, il componente che richiede il gestore dati può richiedere uno dei metodi pubblici di conversione stringa ad oggetto di business, mostrati in Tabella 77.

Tabella 77. Metodi pubblici di conversione stringa ad oggetto di business

Metodo pubblico di conversione stringa ad oggetto di business	**Descrizione**
`getBO(Object` *serializedData*`, Object` *config*`)`	Converte i dati serializzati di `Object` generico in un oggetto di business
`getBO(String` *serializedData*`, Object` *config*`)`	Converte i dati serializzati di `Object` in un oggetto di business
`getBO(InputStream` *serializedData*`, Object` *config*`)`	Converte i dati serializzati di `InputStream` in un oggetto di business
`getBO(byte[]` *serializedData*`, Object` *config*`)`	Converte i dati serializzati di un array di byte in un oggetto di business

fare riferimento anche a

getBO() - public

getBO() - public

Converte i dati serializzati in un oggetto di business.

Syntax

```
public BusinessObjectInterface getBO(Object serializedData,
    Object config);
public BusinessObjectInterface getBO(String serializedData,
    Object config);
public BusinessObjectInterface getBO(InputStream serializedData,
    Object config);
public BusinessObjectInterface getBO(byte[] serializedData,
    Object config);
public void getBO(Object serializedData,
    BusinessObjectInterface theBusObj, Object config);
```

Parametri

serializedData E' un riferimento per i dati serializzati.

theBusObj E' l'oggetto di business da compilare con i dati.

config E' un oggetto facoltativo che contiene informazioni aggiuntive di configurazione per il gestore dati.

Valori di restituzione

I primi quattro moduli restituiscono un oggetto di business compilato con i dati dell'oggetto di immissione dati `Object`, `String`, `InputStream` o con un oggetto array di byte. Il quinto modulo compila l'oggetto di business specificato con i dati provenienti dai dati serializzati.

Eccezioni

`Eccezione`
L'eccezione viene emessa se il metodo non è in grado di convertire i dati serializzati in un oggetto di business.

`NotImplementedException`
L'eccezione viene lanciata se la versione pubblica del metodo `getBO()` non è implementata.

Note

Questo metodo `getBO()` è il metodo pubblico per effettuare la conversione stringa a oggetto di business. La classe base `DataHandler` include i formati astratti di `getBO()` (come descritto nella pagina 205), che devono essere implementati come parte di una classe del gestore dati. Questa versione pubblica di `getBO()` definisce una serie di metodi di utilità che consentono ad un componente (come un connettore o un client di accesso) di specificare gli oggetti *serializedData* come un `Object`, `String` o `InputStream` o come un array di byte. Il metodo converte i dati serializzati specificati in un oggetto `Reader` e quindi richiede uno dei metodi astratti `getBO()` per convertire l'oggetto `Reader` in un oggetto di business.

Il metodo pubblico `getBO()` ha i seguenti moduli:

- Il primo modulo richiede il metodo appropriato `getBO()` in base al tipo dell'oggetto *`serializedData`*. Ad esempio, se il tipo dei dati è `String`, il metodo richiede `getBO(String`*`serializedData`*`, Object`*`config`*`)`.
- Il secondo, il terzo e il quarto modulo creano un oggetto `Reader` da `String`, `InputStream`, o array di byte e richiede il metodo astratto `getBO()` per convertire l'oggetto `Reader` in un oggetto di business.
- Il quinto modulo è un altro programma di utilità che gestisce le richieste `getBO()` quando il richiedente passa in un oggetto di business. Effettua le seguenti attività:
  - determina il tipo di oggetto `serializedData` che viene passato
  - converte l'oggetto in un oggetto `String` o `InputStream`, se appropriato
  - richiede la versione astratta, passando l'oggetto di business oltre ai dati

Per ulteriori informazioni sull'argomento *`config`*, fare riferimento alle relative descrizioni del modulo astratto di `getBO()` (come descritto alla pagina 205).

fare riferimento anche a

getBO() - abstract

---

getBOName()

Ottiene il nome di un oggetto di business in base al contenuto dei dati serializzati.

Syntax

```
public String getBOName(Reader serializedData);
public String getBOName(String serializedData);
public String getBOName(InputStream serializedData);
```

Parametri

serializedData Un riferimento ad un oggetto `Reader` che contiene il messaggio.

Valori di restituzione

Restituisce un oggetto `String` contenente il nome dell'oggetto di business. Se non esiste nessun valore per l'attributo `NameHandlerClass`, questo metodo restituisce `null`.

Eccezioni

Il secondo e il terzo modulo di `getBOName()` possono emettere le seguenti eccezioni:

`MalformedDataException`
L'eccezione viene emessa se i dati serializzati (*serializedData*) non sono nel formato corretto.

`NotImplementedException`
L'eccezione viene emessa se il gestore nome non è implementato.

Note

Il metodo `getBOName()` crea un'istanza di un gestore nome per estrarre il nome della definizione dell'oggetto di business dai dati serializzati. Avvia un'istanza per questo gestore nome in base al valore dell'attributo `NameHandlerClass` del meta oggetto. Il gestore nome crea il nome dell'oggetto di business in base ai contenuti del messaggio.

Il metodo `getBOName()` ha i seguenti formati:

- Il primo modulo restituisce il nome dell'oggetto di business in base all'attributo del meta oggetto `BOPrefix` (se esiste) così come al valore restituito dalla classe `NameHandler`.
- Il secondo modulo crea un oggetto `Reader` da `String` e quindi richiede il primo modulo.
- Il terzo modulo crea un oggetto `Reader` da `InputStream` e quindi richiede il primo modulo.

Attualmente, solo i seguenti gestori dati IBM utilizzano questo metodo:

- gestore dati XML

  Il gestore nome predefinito per il gestore dati XML richiede la classe base `getBOName(Reader`*`data`*`)`. Se il gestore dati non può gestire la richiesta, `<!DOCTYPE Name` viene utilizzato per estrarre il nome base dell'oggetto di business. Il nome finale viene formato nel modo seguente:

  ```
  BOPrefix + "_" + Name.getStreamFromBO()
  ```

- Gestore dati EDI

  Il gestore nome predefinito per il gestore dati EDI ottiene il nome dell'oggetto di business dalla tabella di ricerca del gestore nome EDI.

Per creare il proprio gestore nome, estendere la classe base astratta `NameHandler` e ignorare il metodo `getBOName()`.

Per ulteriori informazioni, consultare “Creazione di un gestore nome personalizzato XML” a pagina 89.

getBooleanOption()

Ottiene il valore dell'opzione di configurazione specificata del gestore dati se contiene dati booleani.

Syntax

```
public boolean getBooleanOption(String name);
```

Parametri

name Nome dell'opzione di configurazione.

Valori di restituzione

Restituisce il valore dell'opzione tipo `Boolean`.

Note

Il metodo `createHandler()` utilizza qualsiasi meta oggetto secondario associato con il gestore dati per inizializzare le relative opzioni di configurazione. E' possibile utilizzare il metodo `getBooleanOption()` per ottenere il valore di una di questa opzioni, sempre che l'opzione contenga un valore booleano.

getByteArrayFromBO()

Serializza un oggetto di business in un array di byte.

Syntax

```
abstract byte[] getByteArrayFromBO(BusinessObjectInterface theBusObj,
      Object config);
```

Parametri

theBusObj Oggetto di business che deve essere convertito in un array di byte.

config Oggetto facoltativo che contiene informazioni aggiuntive di configurazione per il gestore dati.

Valori di restituzione

Un array `byte` contenente dati serializzati che rappresentano l'oggetto di business specificato.

Eccezioni

`Eccezione`
L'eccezione viene lanciata se il metodo non può convertire l'oggetto di business in un array di byte di dati serializzati.

Note

Il metodo `getByteArrayFromBO()` effettua la conversione oggetto di business in per un gestore dati. Converte i dati nell'oggetto di business *`theBusObj`* in un array di byte (un oggetto Java `byte[]`).

Importante: Il metodo `getByteArrayFromBO()` è un metodo astratto che non presenta implementazioni predefinite.Perciò, la classe del gestore dati *deve* implementare questo metodo.

Se è necessario fornire al proprio gestore dati maggiori informazioni di configurazione di quelle incluse nel meta oggetto, è possibile utilizzare l'opzione *config* per passare ad un oggetto che contiene queste informazioni. Ad esempio, *config* potrebbe essere un file maschera o una stringa ad un URL per uno schema che viene utilizzato per creare un documento XML da un oggetto di business.

Se *config* è un tipo di oggetto di business, è necessario implementare il metodo `getByteArrayFromBO()` per richiedere `setupOptions(`*config*`)`. Il metodo `setupOptions()` è definito nella classe base `DataHandler`. Tale metodo utilizza i nomi dell'attributo nell'oggetto di business come nomi della proprietà e i valori predefiniti come valori per quelle proprietà. Imposta i valori delle proprietà di configurazione nell'oggetto per l'utilizzo da parte del gestore dati.

fare riferimento anche a

getBO() - public, getStreamFromBO(), getStringFromBO()

---

getEncoding()

Recupera la codifica carattere che il il gestore dati sta utilizzando.

Syntax

```
public final String getEncoding();
```

Parametri

None.

Valori di restituzione

Un elemento `String` contenente la codifica carattere del gestore dati.

Note

Il metodo `getEncoding()` recupera la codifica carattere del gestore dati. La codifica carattere è parte della locale, che definisce convenzioni culturali per i dati secondo la lingua, il paese (o il territorio).Tale metodo è il metodo accessorio che ottiene il valore di una variabile privata di codifica carattere nella classe `DataHandler`. Questa codifica carattere dovrebbe indicare la codifica carattere dei dati serializzati che il gestore dati sta elaborando.

Questo metodo è utile quando il gestore dati ha bisogno di effettuare un'elaborazione della codifica carattere, come una conversione carattere.

fare riferimento anche a

setEncoding()

---

getLocale()

Recupera la la locale del gestore dati.

Syntax

```
public final Locale getLocale();
```

Parametri

None.

Valori di restituzione

Un oggetto Java `Locale` che descrive la locale per l'ambiente del gestore dati.

Note

Il metodo `getLocale()` recupera la locale del gestore dati, che definisce convenzioni culturali per i dati secondo la lingua, il paese (o il territorio) e una codifica carattere. Questo metodo è il metodo accessorio che recupera il valore di una variabile locale privata nella classe `DataHandler`. Per impostazione predefinita, la locale del gestore dati è la locale del sistema operativo su cui il gestore dati è in esecuzione.

Questo metodo è utile quando il gestore dati ha bisogno di effettuare un elaborazione relativa alla locale.

fare riferimento anche a

setLocale()

---

getOption()

Ottiene il valore dell'opzione di configurazione del gestore dati (se è stata impostata).

Syntax

```
public String getOption(String name);
```

Parametri

name Nome dell'opzione di configurazione.

Valori di restituzione

Un oggetto `String`che contiene il valore dell'opzione.

Note

Il metodo `getOption()` ottiene il valore dell'opzione di configurazione. Se il gestore dati possiede un meta oggetto secondario associato, il metodo `createHandler()` utilizza i valori predefiniti di questi attributi del meta oggetto per inizializzare le relative opzioni di configurazione. E' possibile utilizzare il metodo `getOption()` per ottenere il valore di una di queste opzioni.

fare riferimento anche a

setOption()

getStreamFromBO()

Serializza un oggetto di business in un oggetto `InputStream`.

Syntax

```
abstract InputStream getStreamFromBO(BusinessObjectInterface theBusObj,
      Object config);
```

Parametri

theBusObj Oggetto di business che deve essere convertito in un flusso.

config Oggetto facoltativo che contiene informazioni aggiuntive di configurazione per il gestore dati.

Valori di restituzione

Un oggetto `InputStream` contenete dati serializzati che rappresentano un oggetto di business.

Eccezioni

`Eccezione`
L'eccezione viene lanciata se il metodo non può convertire l'oggetto di business in un flusso di dati serializzati.

Note

Il metodo `getStreamFromBO()` effettua la conversione da oggetto di business a flusso per un gestore dati. Converte i dati nell'oggetto di business *`theBusObj`* in un flusso (un oggetto Java `InputStream`).

Importante: Il metodo `getStreamFromBO()` è un metodo astratto che non ha implementazioni predefinite.Perciò, la classe del gestore dati *deve* implementare questo metodo.

Se è necessario fornire al proprio gestore dati maggiori informazioni di configurazione di quelle incluse nel meta oggetto, è possibile utilizzare l'opzione `config` per passare ad un oggetto che contiene queste informazioni. Ad esempio, *`config`* potrebbe essere un file maschera o una stringa ad un URL per uno schema che viene utilizzato per creare un documento XML da un oggetto di business.

Se `config` è un tipo di oggetto di business, è possibile implementare il metodo `getStreamFromBO()` per richiedere `setupOptions(`*`config`*`)`. Il metodo `setupOptions()` è definito nella classe base `DataHandler`. Tale metodo utilizza i nomi dell'attributo nell'oggetto di business come nomi della proprietà e i valori predefiniti come valori per quelle proprietà. Imposta i valori delle proprietà di configurazione nell'oggetto per l'utilizzo da parte del gestore dati.

fare riferimento anche a

getBO() - public, getByteArrayFromBO(), getStringFromBO()

getStringFromBO()

Serializza un oggetto di business in un oggetto `String`.

Syntax

```
abstract String getStringFromBO(BusinessObjectInterface theBusObj,
      Object config);
```

Parametri

theBusObj Oggetto di business che deve essere convertito in un oggetto `String`.

config Oggetto facoltativo che contiene informazioni aggiuntive di configurazione per il gestore dati.

Valori di restituzione

Un oggetto `String` contenente dati serializzati che rappresentano i dati nell'oggetto di business.

Eccezioni

`Eccezione`

L'eccezione viene lanciata se il metodo non può convertire l'oggetto di business in una stringa di dati serializzati.

Note

Il metodo `getStringFromBO()` effettua la conversione oggetto di business a stringa per un gestore dati. Converte i dati nell'oggetto di business *`theBusObj`* in un oggetto Java `String`.

Importante: Il metodo `getStringFromBO()` è un metodo astratto che non ha implementazioni predefinite.Perciò, la classe del gestore dati *deve* implementare questo metodo.

Se è necessario fornire al proprio gestore dati maggiori informazioni di configurazione di quelle incluse nel meta oggetto, è possibile utilizzare l'opzione `config` per passare ad un oggetto che contiene queste informazioni. Ad esempio, *`config`* potrebbe essere un file maschera o una stringa ad un URL per uno schema che viene utilizzato per creare un documento XML da un oggetto di business.

Se `config` è un tipo di oggetto di business, è possibile implementare il metodo `getStreamFromBO()` per richiedere `setupOptions(`*`config`*`)`. Il metodo `setupOptions()` è definito nella classe base `DataHandler`. Tale metodo utilizza i nomi dell'attributo nell'oggetto di business come nomi della proprietà e i valori predefiniti come valori per quelle proprietà. Imposta i valori delle proprietà di configurazione nell'oggetto per l'utilizzo da parte del gestore dati.

fare riferimento anche a

getBO() - public, getByteArrayFromBO(), getStreamFromBO()

---

setConfigMOName()

Imposta il nome del del gestore dati di livello superiore meta oggetto in una proprietà statica della classe base `DataHandler`.

Syntax

```
public static void setConfigMOName(String name);
```

Parametri

name Stringa contenente il nome del meta oggetto del gestore dati.

Valori di restituzione

None.

Eccezioni

Eccezione
L'eccezione viene lanciata se il metodo imposta il meta oggetto specificato del gestore dati di livello superiore.

Note

Il meta oggetto del gestore dati di livello superiore contiene i tipi MIME supportati e i nomi dei relativi meta oggetti secondari associati. Per ulteriori informazioni sui meta oggetti del gestore dati, fare riferimento a "Configurazione dei gestori dati" a pagina 21.

---

setEncoding()

Imposta la codifica carattere che il gestore dati sta utilizzando.

Syntax

```
public final void setEncoding(String encodingName);
```

Parametri

encodingName
Un oggetto `String` contenente il nuovo valore da assegnare come codifica carattere del gestore dati.

Valori di restituzione

None.

Note

Il metodo `setEncoding()` imposta la codifica carattere del gestore dati. La codifica carattere è parte della locale, che definisce convenzioni culturali per i dati secondo la lingua, il paese (o il territorio). Questo metodo è il metodo accessorio che imposta una variabile privata della codifica carattere nella classe `DataHandler`. Questa codifica carattere dovrebbe indicare la codifica carattere dei dati serializzati che il gestore dati sta elaborando.

Questo metodo è utile quando il gestore dati ha bisogno di effettuare un'elaborazione della codifica carattere, come una conversione carattere.

fare riferimento anche a

getEncoding()

---

setLocale()

Imposta la la locale del gestore dati.

Syntax

```
public final void setLocale(Locale localeObject);
```

Parametri

localeObject
Un oggetto Java `Locale` contenente la nuova locale da assegnare all'ambiente del gestore dati.

Valori di restituzione

None.

Note

Il metodo `setLocale()` imposta la locale del gestore dati, che definisce convenzioni culturali per i dati secondo la lingua, il paese (o il territorio) e una codifica carattere. Questo metodo è il metodo accessorio che imposta una variabile locale privata nella classe `DataHandler`. Questa locale dovrebbe indicare la locale dei dati serializzati che il gestore dati riceve e crea.

Questo metodo è utile quando il gestore dati ha bisogno di modificare la locale del gestore dati. Un utilizzo possibile è quello di specificare una locale differente per i dati serializzati da convertire in un oggetto di business. Richiedere `setLocale()` prima di una richiesta per `getBO()` modifica la locale che `getBO()` utilizza quando associa una locale con l'oggetto di business che crea.

fare riferimento anche a

getLocale()

---

setOption()

Imposta il valore dell'opzione di configurazione specificata del gestore dati.

Syntax

```
public void setOption(String name, String value);
```

Parametri

name Nome dell'opzione di configurazione.

value Valore dell'opzione di configurazione.

Valori di restituzione

None.

Note

Il metodo `setOption()` assegna un nuovo valore all'opzione di configurazione. Se il gestore dati possiede un meta oggetto secondario associato, il metodo `createHandler()` utilizza i valori predefiniti di questi attributi del meta oggetto per inizializzare le relative opzioni di configurazione. Se il gestore dati è richiesto nel contesto di un connettore, tale meta oggetto secondario è nella memoria del processo del connettore. Se si utilizza InterChange Server Express e il gestore dati è richiesto nel contesto di un client di accesso, tale meta oggetto è nella memoria del

processo di InterChange Server Express. E' possibile utilizzare il metodo `setOption()` per ignorare il valore di una di queste opzioni.

Nota: Modificare un opzione di configurazione con `setOption()` imposta il valore dell'attributo del meta oggetto in memoria. La modifica *non* influisce sul valore dell'attributo nel repository.

fare riferimento anche a

getOption()

---

traceWrite()

Richiama la funzione trace-write appropriata per scrivere un messaggio di traccia per il gestore dati.

Syntax

```
public void traceWrite(String message, int level);
```

Parametri

message Il messaggio di testo da utilizzare per il messaggio di traccia.

level Un intero che specifica il livello di traccia per il messaggio. Il livello di traccia è tra 0-5, dove 0 specifica nessuna traccia e 5 specifica il livello massimo di traccia.

Valori di restituzione

None.

Note

Il metodo `traceWrite()` è un metodo wrapper che richiama la funzione trace-write appropriata in base al contesto in cui il gestore dati viene eseguito. Il livello di traccia predefinito è il livello di traccia del connettore. Se si utilizza InterChange Server Express e Il gestore dati viene eseguito nel contesto di Server Access Interface, il metodo `traceWrite()` imposta il sottosistema del livello di traccia prima di richiamare il metodo trace-write.

Appendice. Utilizzare l'ODA XML

Questo capitolo descrive ODA XML, un agente di rilevamento oggetti (ODA) che genera definizioni dell'oggetto di business per documenti XML. Poiché un documento XML può avere il relativo schema definito o da DTD (Document type definition) o da un documento dello schema, l'ODA XML ODA può utilizzare entrambi questi modelli di dati per trovare gli specifici requisiti dell'oggetto di business nel documento XML.

Questo capitolo contiene le sezioni seguenti:

Installazione e utilizzo

Questa sezione discute dei seguenti argomenti:

Installazione dei ODA XML

Per le istruzioni su come installare l'ODA con Express Installer, fare riferimento a *WebSphere Business Integration Server Express - Guida all'installazione per Linux* o *per Windows*.

Una volta terminata l'installazione, i seguenti file vengono installati nella directory del prodotto nel proprio sistema:

- `ODA\XML\XMLODA.jar`
- `ODA\messages\XMLODAAgent.txt`
- `ODA\messages\XMLODAAgent_ll_TT.txt files`(file di messaggio specifici per una lingua (*ll*) ed un paese o un territorio (*TT*))
- `ODA\XML\start_XMLODA.bat` (solo per Windows)
- `ODA/XML/start_XMLODA.sh` (solo per Linux)

Nota: Eccetto dove altrimenti specificato, in questo manuale le barre rovesciate (\) vengono utilizzate come convenzione per i percorsi di directory. Per le installazioni sui sistemi Linux, è necessario sostituire i caratteri barra inversa con le barre (/). Tutti i nomi di percorso del prodotto sono relativi alla directory del sistema in cui viene installato il prodotto.

Prima di utilizzare L'ODA XML

Prima di eseguire l'ODA XML, verificare che il proprio sistema abbia i file richiesti. In particolare, assicurarsi che il file dell'ambiente ODA sia stato installato nella sottodirectory `bin` della directory del prodotto.

Linux

Assicurarsi che il file dell'ambiente ODA, `CWODAEnv.sh`, sia installato nella seguente directory: *`ProductDir`*`/bin`.

Windows

Assicurarsi che il file dell'ambiente ODA, `CWODAEnv.bat`, sia installato nella seguente directory: *`ProductDir`*`\bin`.

Inoltre, è necessario assicurarsi che le variabili siano correttamente impostate nello script di avvio o nel file batch, che esegue l'ODA. Aprire per modificare la shell (`start_XMLODA.sh`) o il file batch (`start_XMLODA.bat`) e confermare che i valori descritti in Tabella 78 sono corretti.

Tabella 78. Variabili di configurazione della shell e del file batch

Variabile	Spiegazione	Esempio
`impostare AGENTNAME`	Nome dell'ODA	`set AGENTNAME=XMLODA`
`impostare AGENT`	Nome del file jar dell'ODA	**Linux:** `impostare AGENT = ${`*`ProductDir`*`}/ODA/XML/XMLODA.jar` **WINDOWS:** `impostare AGENT = %`*`ProductDir`*`%\ODA\XML\XMLODA.jar`
`impostare AGENTCLASS`	Nome della classe Java dell'ODA	`impostare AGENTCLASS=com.crossworlds.oda.xml.XMLAgent`

Dopo aver installato l'ODA XML ed aver impostato le variabili di configurazione nella shell o nel file batch (fare riferimento a Tabella 78), è necessario effettuare le seguenti operazioni per generare oggetti di business:

1. Avviare l'ODA XML.
2. Avviare Business Object Designer Express.
3. Eseguire la procedura guidata a sei fasi in Business Object, un interfaccia GUI che Business Object Designer Express fornisce per configurare ed eseguire un ODA.

Le seguenti sezioni descrivono questi passaggi in dettaglio.

Avviare l'ODA XML

E' possibile avviare l'ODA XML con lo script di avvio appropriato per il proprio sistema.

Linux

```
start_XMLODA.sh
```

Windows

```
start_XMLODA.bat
```

Nota: Windows Installer fornisce scorciatoie per avviare gli ODA che installa.

La configurazione e l'esecuzione dell'ODA XML ODA avvengono utilizzando Business Object Designer Express. La procedura guidata di Business Object, che Business Object Designer Express avvia, localizza ogni ODA dal nome specificato nella variabile AGENTNAME di ciascuno script o file batch. Il nome predefinito ODA per questo connettore è XMLODA.

Eseguire istanze multiple dell'ODA XML

E' possibile eseguire istanze multiple di un ODA XML sia sulla macchina dell'host locale che sulla macchina dell'host remoto. Ogni istanza esegue il programma su di un'unica porta. E' possibile specificare il numero di porta come parte dell'avvio dell'ODA dall'interno di Business Object Designer Express. Figura 44 a pagina 221 illustra la finestra in Business Object Designer Express da cui si seleziona l'ODA da eseguire.

Operazioni con file di traccia e di errore

Per impostazione predefinita, i file di traccia e di errore (l'impostazione predefinita è XMLODAAgent.txt) si trovano nella sottodirectory \ODA\messages, sotto la directory del prodotto. Questi file utilizzano la seguente convenzione di denominazione:

*AgentName*Agent.txt

Se si creano istanze multiple dello script o del file batch ODA e si fornisce un nome unico per ogni ODA rappresentato, è possibile che quelle istanze utilizzino lo stesso file di messaggio. In alternativa, è possibile specificare differenti file di messaggio per ogni istanza ODA specificando i nomi file in odk.dd.xml, che è il *file descrittore di distribuzione dell'ODA* installato con l'ODA XML.

Per specificare differenti file di messaggio per diverse istanze dell'ODA, è possibile copiare il descrittore di distribuzione principale dell'ODA, installato in \ODA\odk.dd.xml e modificare i valori messagefile, tracefile e tracelevel di conseguenza. Il descrittore di distribuzione principale dell'ODA ha il seguente formato e valori predefiniti:

```
<odk>       <startup>             <messagefile
usestandard="true"></messagefile>       </startup>       <diagnostics>
            <tracefile usestandard="true"></tracefile>
<tracelevel canoverride="true">1</tracelevel>       </diagnostics> </odk>
```

Business Object Designer Express presume che ogni file sia stato denominato secondo la convenzione di denominazione. Ad esempio, se la variabile AGENTNAME specifica XMLODA1, lo strumento presume che il nome del file di messaggio associato sia XMLODA1Agent.txt. Perciò, quando Business Object Designer Express fornisce il nome del file per la verifica come parte della configurazione dell'ODA, il nome del file è basato sul nome dell'ODA. Verificare che il file di messaggio predefinito sia denominato correttamente e correggerlo se necessario.

Importante: Tralasciare di specificare correttamente il nome del file di messaggio durante la configurazione dell'ODA causa l'esecuzione senza messaggi. Per ulteriori informazioni sulla specifica del nome del file di messaggio, fare riferimento a Tabella 80 a pagina 222.

Nota: Non è necessario effettuare copie separate del file di messaggio per eseguire istanze multiple dell'ODA. E' possibile specificare lo stesso nome del file di messaggio nei file oda.dd.xml. Ma è necessario specificare distinti nomi del file di traccia, poiché se i messaggi di traccia di ODA differenti sono diretti allo stesso file di traccia, i messaggi saranno indecifrabili.

Durante il processo di configurazione, è necessario specificare:

- Il nome del file in cui l'ODA XML scrive errori e informazioni di traccia
- Il livello di traccia, che va da `0` a `5`.

Tabella 79 descrive questi valori.

Tabella 79. Livelli di traccia

Livello di traccia	Descrizione
0	Registra tutti gli errori
1	Tiene traccia di tutti i messaggi in entrata e in uscita per il metodo
2	Tiene traccia delle proprietà dell'ODA e i relativi valori
3	Tiene traccia dei nomi di tutti gli oggetti di business
4	Tiene traccia dei dettagli di tutti i thread riprodotti
5	• Indica i valori di inizializzazione dell'ODA per tutte le sue proprietà • Tiene traccia di uno status dettagliato di ogni thread che l'ODA XML ha riprodotto • Tiene traccia della definizione dell'oggetto di business dump

Per informazioni su dove configurare questi valori, fare riferimento a Tabella 80 a pagina 222.

Utilizzo di un ODA XML in Business Object Designer Express

Questa sezione descrive come utilizzare Business Object Designer Express per generare definizioni di un oggetto di business utilizzando l'ODA XML. Per informazioni su come avviare Business Object Designer Express, fare riferimento a *Business Object - Guida allo sviluppo*. Business Object Designer Express fornisce una procedura guidata, chiamata procedura guidata di Business Object, che guida l'utente attraverso ognuno di questi passaggi. Dopo aver avviato un ODA, è necessario avviare Business Object Designer Express per ottenere l'accesso alla procedura guidata di Business Object (che configura ed esegue l'ODA). Sono presenti sei passaggi nella procedura guidata di Business Object per generare le definizioni di un oggetto di business utilizzando un ODA.

Dopo aver avviato l'ODA, effettuare le seguenti operazioni per avviare la procedura guidata:

1. Aprire Business Object Designer Express.
2. Dal menu File, selezionare il sottomenu Nuovo Utilizzo ODA....

   La procedura guidata di Business Object visualizza la prima finestra, denominata Selezionare un agente. Figura 44 a pagina 221 illustra questa finestra.

Per selezionare, configurare ed eseguire l'ODA, eseguire i seguenti passaggi:

1. "Selezionare l'ODA"
2. "Specificare le proprietà di configurazione"
3. "Espandere nodi e selezionare elementi XML" a pagina 224
4. "Confermare la selezione degli oggetti" a pagina 225
5. "Generare la definizione di oggetto di business" a pagina 226 e, facoltativamente, "Fornire informazioni aggiuntive" a pagina 227
6. "Salvare la definizione dell'oggetto di business definition" a pagina 229

Selezionare l'ODA

Figura 44 illustra la prima finestra della procedura guidata a sei fasi in Business Object. Da questa finestra, selezionare l'ODA da eseguire.

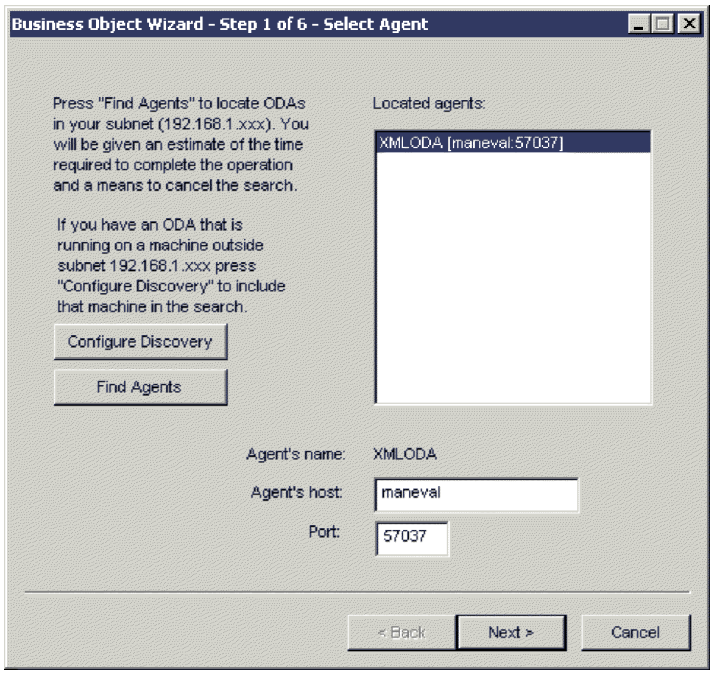

Figura 44. Selezione di un ODA

Per selezionare l'ODA:

1. Fare clic sul pulsante Trova agenti per visualizzare tutti gli ODA registrati o attualmente in esecuzione nel campo Agenti ubicati. In alternativa, è possibile trovare l'ODA utilizzando il suo nome host e il numero della porta.

   Nota: Se la procedura guidata di Business Object non trova l'ODA desiderato, controllare il setup dell'ODA.
2. Selezionare l'ODA desiderato dall'elenco visualizzato.

   La procedura guidata di Business Object visualizza la propria selezione nel campo del nome dell'agente.

Specificare le proprietà di configurazione

La prima volta che la procedura guidata di Business Object comunica con l'ODA XML, richiede di immettere una serie di proprietà di configurazione dell'ODA come mostrato in Figura 45 a pagina 222.

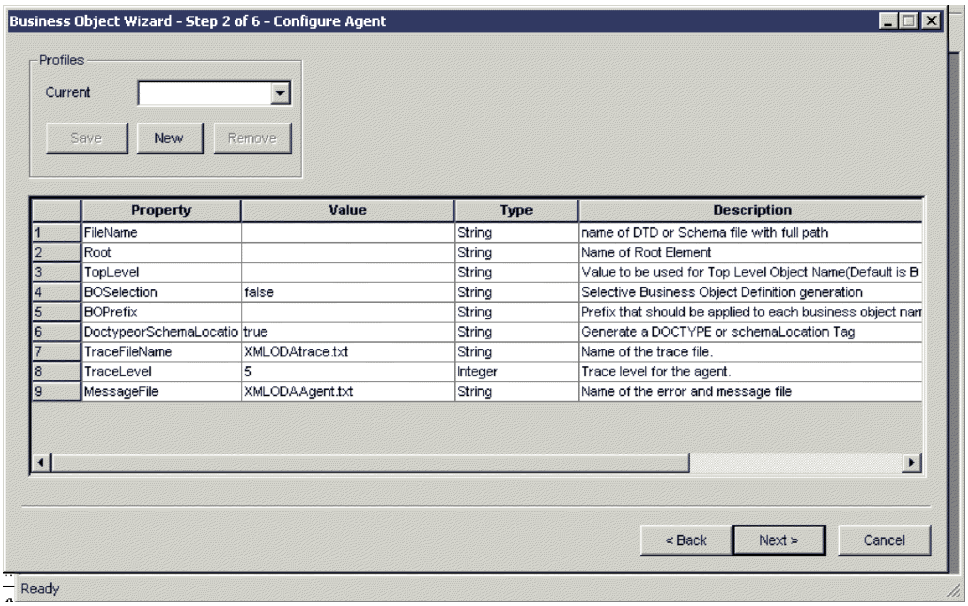

Figura 45. Specificare le proprietà di configurazione dell'ODA

Configurazione delle proprietà dell'XML ODA descritte in Tabella 80.

Tabella 80. proprietà di configurazione dell'ODA XML

Numero di riga	Nome proprietà	Tipo proprietà	Descrizione
1	`FileName`	Stringa	Nome percorso completo di DTD o documento dello schema. Un file DTD deve avere l'estensione `.dtd`; un file del documento dello schema deve avere l'estensione `.xsd`.
2	`Root`	Stringa	Nome dell'elemento XML che deve essere elaborato come elemento root. Se nessun elemento root è specificato, l'ODA XML effettua le seguenti ipotesi: • Se l'ODA sta analizzando un DTD, elabora il primo elemento XML come root. • Se l'ODA sta analizzando un un documento dello schema, elabora il primo elemento globale come root.
3	`TopLevel`	Stringa	Nome da utilizzare per l'oggetto di business di livello superiore che l'ODA genera. L'ODA appone l'oggetto di business di livello superiore con il prefisso dell'oggetto di business (che la proprietà `BOPrefix` specifica), separato da una sottolineatura (_). Se *non* si specifica un nome di livello superiore, l'ODA assegna il nome *`BOPrefix_Root`* (dove *BOPrefix* e *`Root`* sono valori delle proprietà `BOPrefix` e Root) come il nome dell'oggetto di business .

Tabella 80. proprietà di configurazione dell'ODA XML (Continua)

Numero di riga	Nome proprietà	Tipo proprietà	Descrizione
4	`BOSelection`	Stringa	Un valore booleano (`vero` o `falso`) per indicare se l'ODA XML consentirà di selezionare i nomi degli elementi per cui le definizioni dell'oggetto di business devono essere utilizzate. • Se tale proprietà è impostata su `falso`, l'ODA XML consente all'utente di selezionare solo root come elemento e produrrà le definizioni dell'oggetto di per root e tutti i relativi elementi secondari. • Se tale proprietà è impostata su `vero`, l'ODA XML consente all'utente di selezionare qualsiasi elemento e genererà definizioni dell'oggetto di business solo per gli elementi selezionati. L'impostazione predefinita è `falso`.
5	`BOPrefix`	Stringa	Prefisso che l'ODA applica al nome di ogni definizione dell'oggetto di business per un documento XML. Se non si specifica un prefisso dell'oggetto di business, l'ODA *non* appone alcuna stringa al nome della definizione dell'oggetto di business.
6	`DoctypeorSchemaLocation`	Stringa	Un valore booleano (`vero` o `falso`) per indicare se l'ODA XML dovrebbe generare attributi per: • Durante l'elaborazione dei DTD: la tag `DOCTYPE` • Durante l'elaborazione di un documento dello schema: `schemaLocation` e gli attributi `xsi` (nello spazio del nome dell'istanza dello schema XML) L'impostazione predefinita è `vero`.
7	`TraceFileName`	Stringa	Il nome del percorso completo del file in cui l'ODA XML scrive le informazioni di traccia. Se il file non esiste, ODA XML lo crea nella directory specificata. Se il file già esiste, ODA XML lo aggiunge. Per impostazione, l'ODA XML crea un file di traccia denominato `XMLODAtrace.txt` nella sottodirectory `ODA\XML` della directory del prodotto. Utilizzare questa proprietà per specificare un nome differente per il file di traccia.
8	`TraceLevel`	Valore numerico intero	Livello di traccia abilitato per l'ODA XML. I valori validi sono tra zero e cinque (0-5). LA proprietà fa riferimento ad un valore pari a 5 (livello massimo di traccia abilitato). Per ulteriori informazioni, consultare "Operazioni con file di traccia e di errore" a pagina 219.
9	`MessageFile`	Stringa	Nome percorso completo del file di messaggio ed errore. Per impostazione predefinita, l'ODA XML crea un file di messaggio e di errore denominato `XMLODAAgent.txt`. **Importante**: Il file di messaggio e di errore *deve* trovarsi nella sottodirectory `ODA\messages` della directory del prodotto. Utilizzare questa proprietà per verificare o specificare un file esistente.
10	`Bidi.Application`	Stringa	Definisce il formato BiDi del DTD dell'applicazione esterna secondo le specifiche in Tabella 81 a pagina 231.

Importante: Correggere il nome del file di messaggio se il valore predefinito visualizzato in Business Object Designer Express rappresenta un file non esistente. Se il nome non è corretto quando ci si sposta da questa finestra, Business Object Designer Express visualizza un messaggio di errore nella finestra da cui l'ODA è stato lanciato. Questo messaggio non compare in Business Object Designer Express. Tralasciare di specificare un file di messaggio valido comporta che l'ODA viene eseguito senza messaggi.

E' possibile salvare queste informazioni in un profilo denominato in modo da non dover reinmetterle ogni volta che si utilizza l'ODA XML. Per ulteriori informazioni su come specificare un profilo ODA, fare riferimento a *Business Object - Guida allo sviluppo*.

Espandere nodi e selezionare elementi XML

Business Object Designer Express utilizza le proprietà configurate nel passaggio precedente per connettere lo strumento allo schema XML specificato (DTD o documento dello schema). Dopo la connessione, Business Object Designer Express visualizza un albero i cui nodi rappresentano tutti gli elementi XML definiti nello schema XML.

e' possibile espandere l'elemento XML di livello superiore per visualizzare l'intera rappresentazione gerarchica. Per ogni elemento XML, ODA XML crea una definizione dell'oggetto di business secondaria.

Figura 46 a pagina 225 illustra questa finestra con alcuni elementi XML espansi.

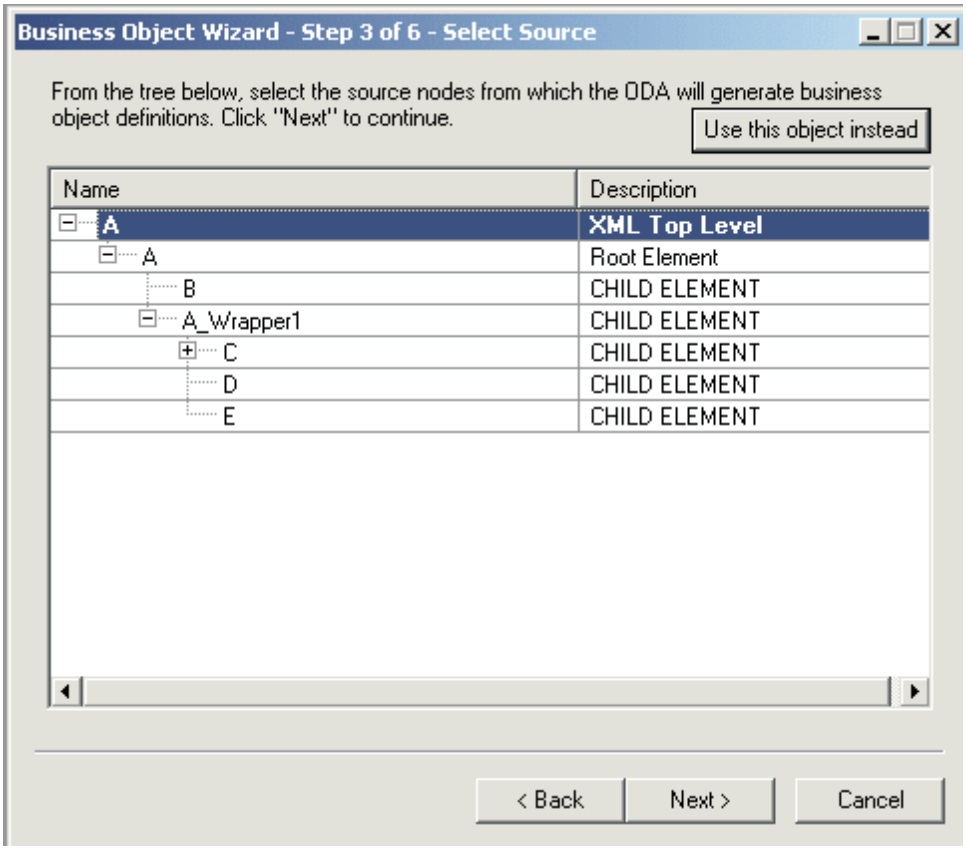

Figura 46. Albero degli elementi XML con i nodi espansi

Selezionare tutti gli elementi XML e fare clic su Avanti.

Confermare la selezione degli oggetti

Dopo aver identificato tutti gli elementi XML che devono essere associati con le definizioni dell'oggetto di business generate, Business Object Designer Express visualizza la finestra con solo gli oggetti selezionati. Figura 47 a pagina 226 illustra questa finestra.

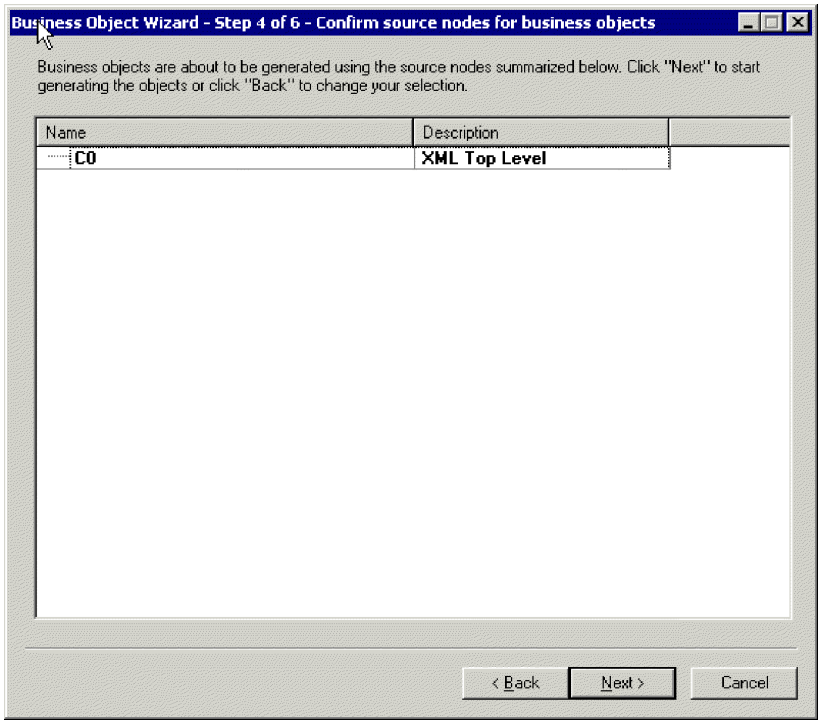

Figura 47. Conferma della selezione degli oggetti

Questa finestra fornisce le seguenti opzioni:

- Per confermare la selezione, fare clic su Avanti.
- Se la selezione non è corretta, fare clic su Indietro per tornare alla finestra precedente ed effettuare i necessari cambiamenti. Una volta che la selezione è corretta, fare clic su Avanti.

Generare la definizione di oggetto di business

Dopo aver confermato gli elementi XML, la seguente finestra avvisa che Business Object Designer Express sta generando la definizione di oggetto di business. Se un grande numero di Interfacce componente è stato selezionato, questo passaggio della generazione può richiedere tempo.

Figura 48 a pagina 227 illustra questa finestra.

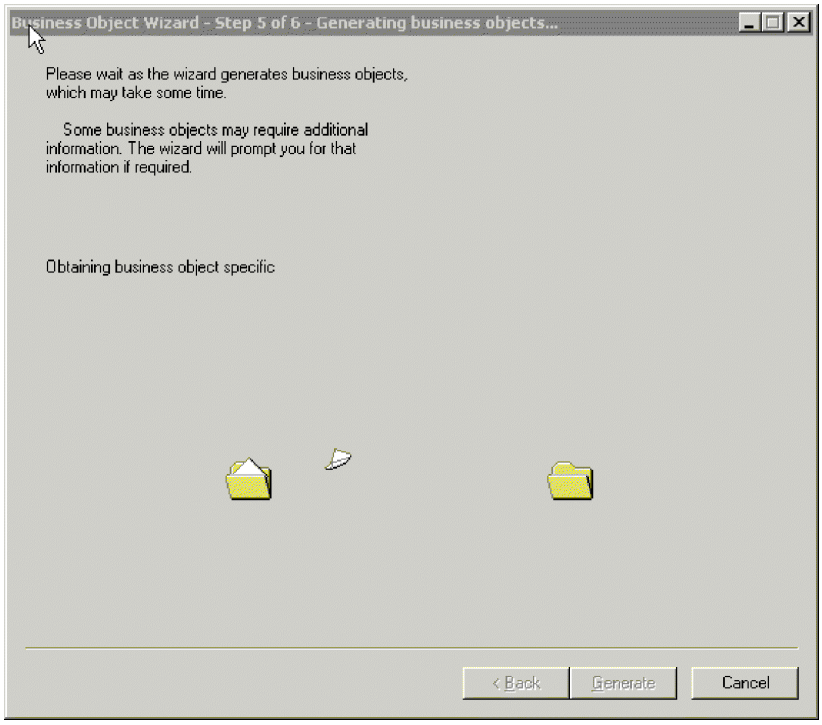

Figura 48. Generare le definizioni di un oggetto di business

L'ODA XML genera il nome per una definizione di oggetto di business dalle seguenti informazioni:

- Il valore della proprietà di configurazione `BOPrefix` dell'ODA
- Il valore della proprietà di configurazione `TopLevel` dell'ODA
- Il nome dell'elemento XML che la definizione dell'oggetto di business rappresenta

L'ODA separa ognuno di questi valori con un carattere sottolineatura (_). Perciò, il nome che genera avrà il seguente formato:

BOPrefix_TopLevel_XMLelement

Fornire informazioni aggiuntive

Poiché l'ODA XML necessita di ulteriori informazioni riguardo ai verbi, Business Object Designer Express visualizza la finestra delle proprietà di BO, che richiederà le informazioni. Figura 49 a pagina 228 illustra questa finestra.

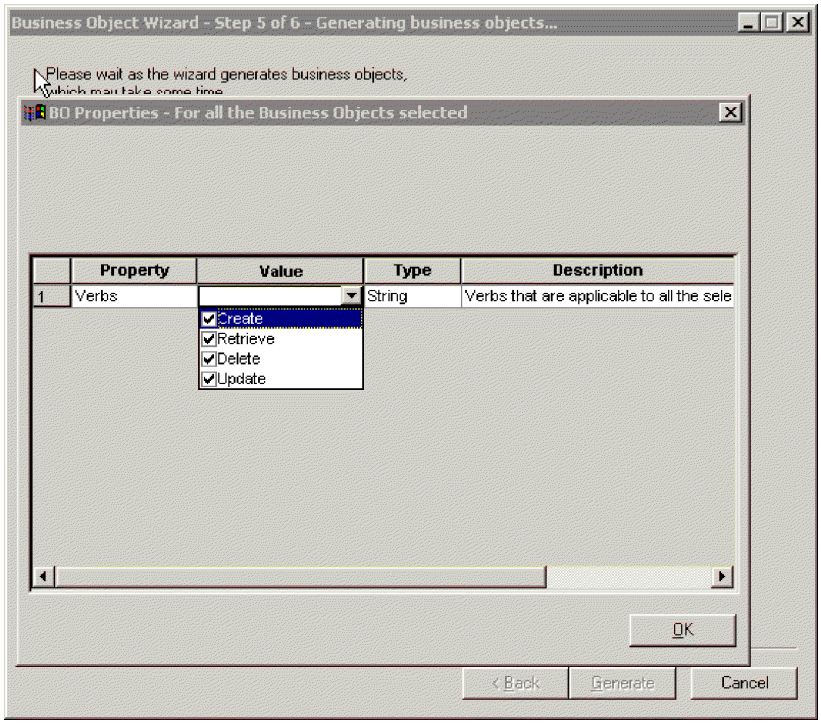

Figura 49. Fornire informazioni aggiuntive - verbi

Nella finestra delle proprietà di BO, immettere o modificare le informazioni del verbo. Fare clic nel campo *Valore* e selezionare uno o più verbi dal menu a comparsa. Questi saranno i verbi supportati dall'oggetto di business.

Nota: Se un campo nella finestra delle proprietà di BO presenta valori multipli, il campo sarà vuoto quando la casella sarà visualizzata per la prima volta. Fare clic nel campo per visualizzare un elenco a discesa dei relativi valori.

Se il proprio documento XML presenta un documento dello schema che contiene un elemento `anyAttribute`, l'ODA XML visualizza una finestra delle proprietà di BO aggiuntiva, come mostrato in Figura 50.

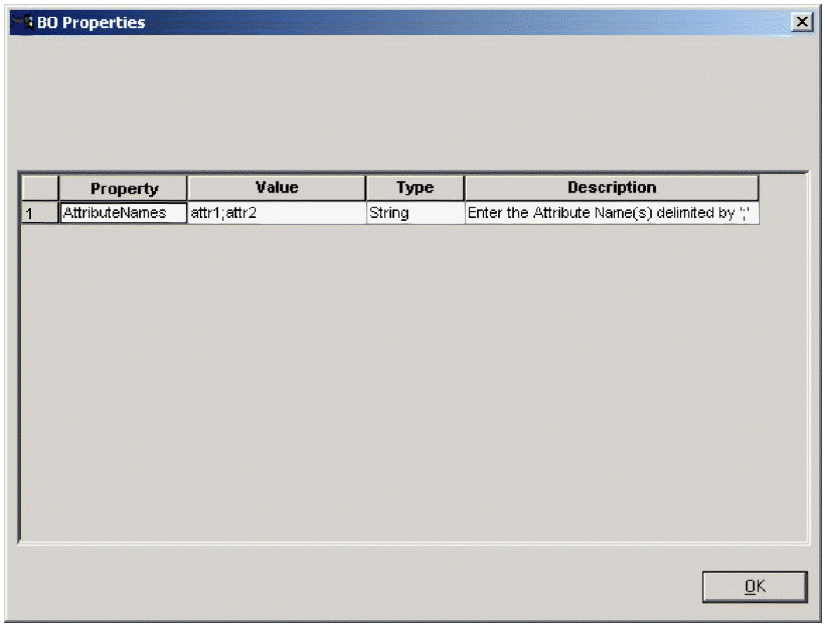

Figura 50. Fornire informazioni aggiuntive - nomi dell'attributo

In questa finestra delle proprietà di BO, immettere i nomi degli attributi dell'oggetto di business che se desidera che l'ODA XML crei. Separare ogni attributo con un punto e virgola (;). Per ulteriori informazioni su `anyAttribute`, fare riferimento a "Strutture del documento dello schema supportate" a pagina 80.

Salvare la definizione dell'oggetto di business definition

Dopo aver fornito tutte le informazioni richieste nella finestra delle proprietà di BO e avere fatto clic su OK, Business Object Designer Express visualizzerà la finestra finale nella procedura guidata. In questa finestra, è possibile effettuare ogni operazione seguente:

- Salvare la definizione di oggetto di business sul server.
- Salvare la definizione di oggetto di business su un file.
- Aprire la definizione di oggetto di business per modificarla in Business Object Designer Express.

Per ulteriori informazioni e per effettuare altre modifiche, fare riferimento a *Business Object - Guida allo sviluppo*.

Figura 51 a pagina 230 illustra questa finestra.

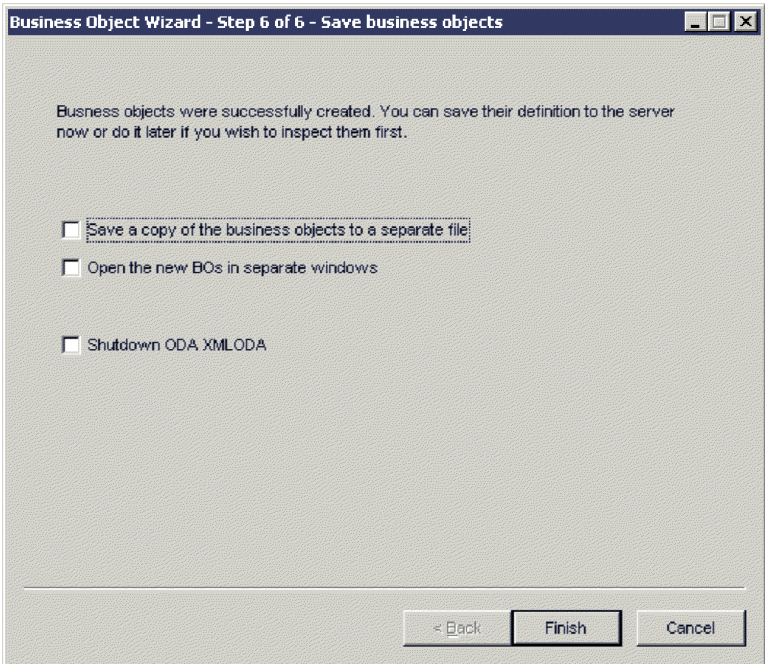

Figura 51. Salvare la definizione di oggetto di business

Contenuti della definizione di un oggetto di generato

La definizione dell'oggetto di business che l'ODA XML genera per gli attributi, i verbi e le informazioni specifiche sull'applicazione come descritto nelle sezioni seguenti:

Componenti della definizione di un oggetto di business	**Per ulteriori informazioni**
Attributi	"Struttura oggetto di business" a pagina 33 "Proprietà dell'attributo oggetto di business" a pagina 34
Informazioni specifiche sull'applicazione	"informazioni specifiche sull'applicazione" a pagina 37
Istruzioni	"Verbi dell'oggetto di business" a pagina 38

Nota: Nelle precedenti versioni dell'ODA XML venivano generate definizioni dell'oggetto di business principali che designavano l'attributo ObjectEventId come chiave. Business Object Designer Express non consente più alle definizioni di un oggetto di business di specificare ObjectEventId come un attributo chiave. Perciò, l'ODA XML ora effettua passaggi speciali per evitare questo comportamento. Questo nuovo comportamento richiede che venga modifica la definizione dell'oggetto di business generata per designare un attributo chiave. Per ulteriori informazioni, consultare "Proprietà degli attributi Chiave e Chiave esterna" a pagina 35.

Modifica delle informazioni nella definizione di un oggetto di business

Può essere necessario modificare le informazioni nella definizione dell'oggetto di business che l'ODA XML crea. Ad esempio, è necessario rimuovere manualmente gli attributi indesiderati ed aggiungere le tag richieste per le informazioni specifiche sull'applicazione dell'attributo. Per esaminare o modificare la definizione

dell'oggetto di business, è possibile utilizzare Business Object Designer Express o un editor di testo. Per caricare nuovamente una definizione modificata nel repository, è possibile utilizzare Business Object Designer Express. In alternativa, è possibile utilizzare il comando `repos_copy` per caricare la definizione nel repository.

Abilitazione del supporto bidirezionale in ODA XML

L'ODA XML è stato globalizzato per gestire schemi XML e DTD che utilizzano serie di caratteri bidirezionali. Il gestore dati XML supporta ora formati di codifica dei caratteri bidirezionali. Per visualizzare il testo da lingue differenti, l'ODA XML utilizza i file di risorsa che possono essere tradotti in varie lingue.

Per abilitare questo supporto, l'ODA XML fa assegnamento sulla proprietà `BiDi.Application` per specificare il formato bidirezionale dello schema XML o del DTD. Il valore di `BiDi.Application` sarà una sequenza di lettere che specificano vari aspetti del formato del testo bidirezionale. Tabella 81 definisce questi valori letterali.

Tabella 81. Specificazione formato BiDi

Posizione lettera	Significato lettera	Valore possibile	Valore predefinito	Significato
1	Tipo	I	I	Implicito
		V		Logico
2	Direzione	L	L	Da sinistra a destra
		R		Da destra a sinistra
3	Cambiamento simmetrico	Y	Y	Cambiamento simmetrico attivo
		N		Cambiamento simmetrico disattivo
4	Elaborazione	S	N	Il testo è elaborato
		N		Il testo non è elaborato
5	Elaborazione numerica	H	N	Hindi (nazionale)
		C		Contestuale
		N		Nominale

Informazioni particolari

Queste informazioni sono state sviluppate per prodotti e servizi offerti negli Stati Uniti. È possibile che negli altri paesi l'IBM non offra i prodotti, le funzioni o i servizi illustrati in questo documento. Consultare il rappresentante locale IBM per informazioni sui prodotti e sui servizi disponibili attualmente nel proprio paese. Qualunque riferimento relativo a prodotti, programmi o servizi IBM non implica che solo quei prodotti, programmi o servizi IBM possano essere utilizzati. In sostituzione a quelli forniti dall'IBM possono essere utilizzati prodotti, programmi o servizi funzionalmente equivalenti che non comportino violazione dei diritti di proprietà intellettuale e di altri diritti dell'IBM. È comunque responsabilità dell'utente valutare e verificare la possibilità di utilizzare altri programmi e/o prodotti, fatta eccezione per quelli espressamente indicati dall'IBM. L'IBM può avere brevetti o domande di brevetto in corso relativi a quanto trattato nel presente documento. La fornitura di questa pubblicazione non implica la concessione di alcuna licenza su di essi. E' possibile inviare domande di licenza, per iscritto a:

IBM Director of Commercial Relations
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
Deutschland

Per domande di autorizzazioni relative a informazioni DBCS, contattare IBM Intellectual Property Department nel proprio paese oppure inviare le domande a:

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

Il seguente paragrafo non è valido per il Regno Unito o per tutti i paesi le cui leggi nazionali siano in contrasto con le disposizioni in esso contenute: L'INTERNATIONAL BUSINESS MACHINES CORPORATION FORNISCE QUESTA PUBBLICAZIONE "NELLO STATO IN CUI SI TROVA", SENZA ALCUNA GARANZIA, ESPLICITA O IMPLICITA, IVI INCLUSE EVENTUALI GARANZIE DI COMMERCIABILITA' ED IDONEITA' AD UNO SCOPO PARTICOLARE. Alcuni stati non consentono la rinuncia a garanzie esplicite o implicite in determinate transazioni; quindi la presente dichiarazione potrebbe essere non essere a voi applicabile. Questa pubblicazione potrebbe contenere imprecisioni tecniche o errori tipografici. Le informazioni incluse in questo documento vengono modificate su base periodica; tali modifiche verranno incorporate nelle nuove edizioni della pubblicazione. L'IBM si riserva il diritto di apportare miglioramenti e/o modifiche al prodotto o al programma descritto nel manuale in qualsiasi momento e senza preavviso. Tutti i riferimenti a siti Web non dell'IBM contenuti in questo documento sono forniti solo per consultazione e non rappresentano in alcun modo un'approvazione di tali siti Web. I materiali disponibile presso i siti Web non fanno parte di questo prodotto e l'utilizzo di questi è a discrezione dell'utente. Tutti i commenti e i suggerimenti inviati potranno essere utilizzati liberamente dall'IBM e diventeranno esclusiva della stessa. Coloro che detengono la licenza su questo programma e desiderano avere informazioni su di esso allo scopo di consentire: (i) uno scambio di informazioni tra programmi indipendenti ed altri (compreso questo) e (ii) l'uso reciproco di tali informazioni, dovrebbero rivolgersi a:

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

Tali informazioni possono essere disponibili, in base ad appropriate clausole e condizioni, includendo in alcuni casi, il pagamento di una tassa. Il programma su licenza descritto in questo manuale e tutto il materiale su licenza ad esso relativo sono forniti dall'IBM nel rispetto delle condizioni previste dalla licenza d'uso. Tutti i dati contenuti in questa pubblicazione sono stati determinati in ambiente controllato. Pertanto, i risultati ottenuti in ambienti operativi diversi possono variare in modo considerevole. Alcune misurazioni potrebbero essere state fatte su sistemi di livello di sviluppo per cui non si garantisce che queste saranno uguali su tutti i sistemi disponibili. Inoltre, alcune misurazioni possono essere state stimate tramite estrapolazione. I risultati attuali possono quindi variare. Gli utenti del presente documento dovranno verificare i dati applicabili per i propri ambienti specifici. Le informazioni relative a prodotti non-IBM sono state ottenute dai fornitori di tali prodotti. L'IBM non ha verificato tali prodotti e, pertanto, non può garantirne l'accuratezza delle prestazioni. Eventuali commenti relativi alle prestazioni dei prodotti non IBM devono essere indirizzati ai fornitori di tali prodotti. Tutti le dichiarazioni riguardanti la direzione o le decisioni future di IBM sono soggette a variazione o ritiro senza preavviso e costituiscono solo degli obiettivi. Questa pubblicazione contiene esempi di dati e prospetti utilizzati quotidianamente nelle operazioni aziendali. Pertanto, può contenere nomi di persone, società, marchi e prodotti. Tutti i nomi contenuti nella pubblicazione sono fittizi e ogni riferimento a nomi e indirizzi reali è puramente casuale. LICENZA DI COPYRIGHT: Queste informazioni contengono programmi applicativi di esempio in lingua originale, che illustrano le tecniche di programmazione su diverse piatteforme operative. E' possibile copiare, modificare e distribuire queste esempi di programmi sotto qualsiasi forma senza alcun pagamento alla IBM, allo scopo di sviluppare, utilizzare, commercializzare o distribuire i programmi applicativi in modo conforme alle API (Application Programming Interface) a seconda della piattaforma operativa per cui gli esempi dei programmi sono stati scritti. Questi esempi non sono stati testati approfonditamente tenendo conto di tutte le condizioni possibili. La IBM, quindi, non può garantire o assicurare l'affidabilità, la praticità o il funzionamento di questi programmi. Se questa pubblicazione viene visualizzata in formato elettronico, è possibile che le fotografie e le illustrazioni a colori non vengano visualizzate.

Informazioni sull'interfaccia di programmazione

Le informazioni sull'interfaccia di programmazione, se fornite, sono designate per creare il software applicativo utilizzando questo programma. Le interfacce di programmazione di utilizzo generale consentono all'utente di scrivere il software di applicazione che ottiene i servizi di questi strumenti del programma'. Tuttavia, queste informazioni possono contenere anche diagnosi, modifiche ed informazioni sull'ottimizzazione. Le informazioni sulle diagnosi, le modifiche e sull'ottimizzazione vengono fornite come supporto per l'esecuzione del debug del software applicativo.

Avvertenza: Non utilizzare queste informazioni sulle diagnosi, le modifiche e sull'ottimizzazione come un'interfaccia di programmazione poichè sono soggette a cambiamenti.

Marchi e marchi di servizio

I seguenti termini sono marchi della International Business Machines Corporation negli Stati Uniti e/o in altri paesi: i5/OS
IBM
il logo IBM
AIX
CICS
CrossWorlds
DB2
DB2 Universal Database
Domino
IMS
Informix
iSeries
Lotus
Lotus Notes
MQIntegrator
MQSeries
MVS
OS/400
Passport Advantage
SupportPac
WebSphere
z/OS
Microsoft, Windows, Windows NT e il logo Windows sono marchi di Microsoft Corporation negli Stati Uniti e/o in altri paesi. MMX, Pentium e ProShare sono marchi di Intel Corporation negli Stati Uniti e/o in altri paesi. Java e tutti i marchi basati su Java sono marchi di Sun Microsystems, Inc. negli Stati Uniti e/o in altri paesi. Linux è un marchio della Linus Torvalds negli Stati Uniti e/o negli altri paesi. Altri nomi di società, di servizi e prodotti possono essere marchi o marchi di servizi di altre società.

WebSphere Business Integration Server Express e Express Plus includono il software sviluppato da Eclipse Project (http://www.eclipse.org/).

WebSphere Business Integration Server Express, versione 4.4, e WebSphere Business Integration Server Express Plus, versione 4.4

Indice analitico

E

F

G

I

J

L

M

O

P

S

T

U

V