Borland AppServer™ 6.7

Schemas

Borland Software Corporation
20450 Stevens Creek Blvd., Suite 800
Cupertino, CA 95014 USA
www.borland.com

BAS67SCHEMA
2006 年 12 月

Borland®

目次

第 1 章

Borland AppServer の概要

Borland AppServer（AppServer）は、企業環境において分散エンタープライズ アプリケーションの開発、デプロイメント、管理を行うための、サービスやツールのセットです。

AppServer は J2EE 1.4 標準の先進実装製品であり、、EJB 2.1、JMS 1.1、Servlet 2.4、JSP 2.0、CORBA 2.6、XML、SOAP などの最新の業界標準技術をサポートします。ボーランドは、2 つのバージョンの AppServer を提供しており、これには、「Java メッセージング サービス（JMS）」に対する最先端のエンタープライズ メッセージング ソリューション（Tibco と OpenJMS）がそれぞれ同梱されています。ユーザーは、AppServer で必要とする機能やサービスのレベルを選択することができ、それらを変更する必要がある場合には、ライセンスをアップグレードすることにより容易に対応できます。

AppServer を利用することにより、J2EE 1.4 プラットフォーム標準を実装した分散 Java/CORBA アプリケーションを安全にデプロイし、さまざまな側面から管理することができます。

AppServer では、インストールごとのサーバー インスタンスの数は無制限です。そのため、同時接続ユーザーの数は無制限です。

AppServer は次のコンポーネントを備えています。

- J2EE 1.4 の実装。
- Apache Web Server バージョン 2.2.3。
- Borland Security。AppServer のセキュリティのためのフレームワークを提供します。
- 先進の集中管理型 JMS 管理ソリューション（Tibco および OpenJMS）。AppServer に同梱されています。
- 分散コンポーネントのための強力な管理ツール群。AppServer の外部で開発されたアプリケーションも含まれます。

AppServer の機能

AppServer では次の機能が提供されます：

- BAS プラットフォームに対するサポート（AppServer に対してサポートされているプラットフォームのリストについては、http://support.borland.com/kbcategory.jspa?categoryID=389 を参照してください）。
- クラスタリング トポロジーに対する完全サポート。
- VisiBroker ORB インフラストラクチャとのシームレスな統合。
- Borland JBuilder 統合開発環境（IDE）との統合。
- 他のボーランド製品（Borland Optimizeit Profiler や ServerTrace など）との統合の強化。
- AppServer により、既存のアプリケーションを Web サービスとして公開したり、新しいアプリケーションや追加 Web サービスと統合することができます。Borland Web サービスは、Apache Axis 1.2 テクノロジー（SOAP 1.2 をサポートする次世代 Apache SOAP サーバー）をベースとしています。

Borland AppServer のドキュメント

AppServer 関連のドキュメントには次のものがあります：

- **『Borland AppServer インストール ガイド』**：AppServer をネットワーク上にインストールする方法について説明されています。これは、Windows、UNIX の各オペレーティング システムに精通しているシステム管理者の方を対象に書かれています。
- **『Borland AppServer 開発者ガイド』**：運用環境における分散オブジェクト ベース アプリケーションのパッケージング、デプロイメント、管理についての詳細情報が記載されています。
- **『Borland 管理コンソール ユーザーズ ガイド』**：Borland 管理コンソール GUI の使用方法についての情報が記載されています。
- **『Borland セキュリティ ガイド』**：VisiSecure for VisiBroker for Java や VisiSecure for VisiBroker for C++ など、AppServer のセキュリティを確保するためのボーランドのフレームワークについて説明されています。
- **『Borland VisiBroker for Java 開発者ガイド』**：Java による VisiBroker アプリケーションの開発方法について説明されています。本書により VisiBroker ORB の設定と管理、プログラミング ツールの使用方法に精通できるよう、記載されています。また、IDL コンパイラ、スマート エージェント、ロケーション サービス、ネーミング サービス、イベント サービス、オブジェクト アクティベーション デーモン（OAD）、サービス品質（QoS: Quality of Service）、インターフェース リポジトリについても説明されています。
- **『Borland VisiBroker VisiTransact ガイド』**：OMG オブジェクト トランザクション サービス仕様に対するボーランドの実装、および、ボーランドのトランザクション サービス統合コンポーネントについて説明されています。

通常、ドキュメントにアクセスするには、AppServer 製品と共にインストールされるヘルプ ビューアを使用します。ユーザーは、スタンドアロンのヘルプ ビューアから、もしくは AppServer GUI ツールから、ヘルプを参照することができます。どちらの場合も、独立したウィンドウ内にヘルプ ビューアが起動されるため、ナビゲーション ペインを利用できるだけでなく、ナビゲーションや印刷のためのヘルプ ビューアのメイン ツールバーも利用することができます。ヘルプ ビューアのナビゲーション ペインには、すべての AppServer ドキュメントや参考ドキュメントの目次、インデックス、包括的な検索を実行できるページがあります。

PDF 形式の『Borland AppServer 開発者ガイド』や『Borland 管理コンソール ユーザーズ ガイド』は、http://info.borland.com/techpubs/appserver より入手可能です。

AppServer オンライン ヘルプ トピックへのアクセス

オンライン ヘルプにアクセスするには（次のいずれかの方法を利用）：

Windows の場合

- [スタート｜すべてのプログラム｜ Borland AppServer ｜ Help Topics] を選択。
- または、Web ブラウザを起動し、<AppServer_Home>/doc/index.html を開く。

UNIX の場合

- Web ブラウザを起動し、<AppServer_Home>/doc/index.html を開く。

AppServer GUI ツールから AppServer オンライン ヘルプ トピックへのアクセス

AppServer GUI ツールからオンライン ヘルプにアクセスするには（次のいずれかの方法を利用）：

- Borland 管理コンソールから、[Help ｜ Help Topics] を選択。
- Borland デプロイメント ディスクリプタ エディタ（DDEditor）から、[Help ｜ Help Topics] を選択。

ドキュメント表記規則

AppServer のドキュメントでは、文中の特定の部分を表すために、次の表に示す書体や記号を使用しています：

表記規則	用途
ボールド	新規の用語およびドキュメント名に使用されます。
`computer`	ユーザーやアプリケーションが提供する情報、サンプル コマンドライン、およびコードです。
`bold computer`	本文では、ユーザーが入力する情報を示します。サンプル コードでは、重要な文章を強調表示します。
[]	省略可能な項目であることを示します。
…	直前の引数が繰り返し可能であることを示します。
|	二者択一であることを示します。

プラットフォームの表記規則

AppServer のドキュメントでは、プラットフォーム固有の情報を表すために、次の記号を使用しています：

記号	意味
Windows	サポートされているすべての Windows プラットフォーム
Win2003	Windows 2003 のみ
WinXP	Windows XP のみ
Win2000	Windows 2000 のみ
UNIX	サポートされているすべての UNIX プラットフォーム
Solaris	Solaris のみ

Borland サポートへのお問い合わせ

ボーランド社は各種のサポート オプションを提供しています。それらには、インターネット上からの無償サービスもあり、大規模な情報データベースを検索したり、他のボーランド製品ユーザーからの情報を得たりすることが可能です。また、ボーランド製品のインストールに関するサポートから、有償のコンサルタント レベルのサポート、および高レベルなアシスタンスに至るまでの複数のカテゴリから、電話サポートの種類を選択できます。

ボーランドのサポート サービスについての詳細情報の入手や、実際にテクニカル サポートへお問い合わせいただくには、Web サイト http://support.borland.com を参照の上、製品をお使いになっている地域を選択してください。

ボーランド社のサポートへの連絡にあたっては、次の情報をご用意ください。

- 名前
- 会社名およびサイト ID
- 電話番号
- ユーザー ID（米国のみ）
- オペレーティング システムおよびバージョン
- ボーランド製品名およびバージョン
- 適用済みのパッチまたはサービス パック
- クライアントの言語とそのバージョン（使用している場合）
- データベースとそのバージョン（使用している場合）
- 発生した問題の詳細な内容と経緯
- 問題を示すログファイル
- 発生したエラー メッセージまたは例外の詳細な内容

オンライン リソース

ネットワーク上の次のサイトから情報を得ることができます。

ワールド ワイド ウェブ： http://www.borland.com

オンライン サポート： http://support.borland.com （ユーザー ID が必要）

ワールド ワイド ウェブ

http://www.borland.com は、定期的にご確認ください。AppServer 製品チームによる、ホワイト ペーパー、競合製品の分析、FAQ への回答、サンプル アプリケーション、更新ソフトウェア、更新ドキュメント、および新旧製品に関する情報が掲載されています。

特に、次の URL を確認されることをお勧めします：

- http://www.borland.com/downloads/download_appserver.html （AppServer ソフトウェアおよびその他のファイル）
- http://support.borland.com （AppServer FAQ）

Borland ニュースグループ

AppServer を対象とした数多くのスレッド化されたディスカッション グループに参加することができます。Enterprise Server やその他のボーランド製品に関する、ユーザー主体のニュースグループへ参加するには、http://www.borland.com/newsgroups を参照してください。

メモ これらのニュースグループはユーザーによって管理されているものであり、ボーランド社の公式サイトではありません。

第 2 章

アプリケーション モジュール：application-borland.xml

XSD: application_1_4-borland.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://support.borland.com/appserver/xml/ns/j2ee"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:borl="http://
support.borland.com/appserver/xml/ns/j2ee" xmlns="http://www.w3.org/2001/
XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"
version="2.4">
 <element name="application">
  <complexType>
   <sequence>
    <element name="module" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <choice>
        <element name="connector" type="xsd:string"/>
        <element name="ejb" type="xsd:string"/>
        <element name="java" type="xsd:string"/>
        <element name="web">
         <complexType>
          <sequence>
           <element name="web-uri" type="xsd:string"/>
          </sequence>
         </complexType>
        </element>
       </choice>
       <element name="hosts" type="xsd:string" minOccurs="0"/>
      </sequence>
     </complexType>
    </element>
    <element name="env-def" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
    <element name="property" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="prop-name" type="xsd:string"/>
       <element name="prop-type" type="xsd:string" minOccurs="0"/>
       <element name="prop-value" type="xsd:string"/>
```

```
    </sequence>
   </complexType>
  </element>
  <element name="authorization-domain" type="xsd:string" minOccurs="0"/>
  <element name="security-role" minOccurs="0" maxOccurs="unbounded">
   <complexType>
    <sequence>
     <element name="role-name" type="xsd:string"/>
     <element name="deployment-role" type="xsd:string" minOccurs="0"/>
    </sequence>
   </complexType>
  </element>
 </sequence>
</complexType>
</element>
</schema>
```

<application> 要素

```
<xsd:element name="application">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="module" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:choice>
       <xsd:element name="connector" type="xsd:string"/>
       <xsd:element name="ejb" type="xsd:string"/>
       <xsd:element name="java" type="xsd:string"/>
       <xsd:element name="web">
        <xsd:complexType>
         <xsd:sequence>
          <xsd:element name="web-uri" type="xsd:string"/>
         </xsd:sequence>
        </xsd:complexType>
       </xsd:element>
      </xsd:choice>
      <xsd:element name="hosts" type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="env-def" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
   <xsd:element name="property" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="prop-name" type="xsd:string"/>
      <xsd:element name="prop-type" type="xsd:string" minOccurs="0"/>
      <xsd:element name="prop-value" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="authorization-domain" type="xsd:string" minOccurs="0"/>
   <xsd:element name="security-role" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="role-name" type="xsd:string"/>
      <xsd:element name="deployment-role" type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
```

```
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

この要素は、`application-borland.xml` のルート ノードで、Borland AppServer パーティションにデプロイされたアプリケーションの EAR モジュールに必要なランタイム情報を定義します。このノードには、`module`、`env-def`、`property`、`authorization-domain`、および `security-role` サブ要素が、0 個以上、含まれています。

サンプル

```
<application>
    <module>
        <ejb>my-ejb.jar</ejb>
    </module>
    <module>
        <web>
            <web-uri>myweb.war</web-uri>
        </web>
    </module>
    <authorization-domain>default</authorization-domain>
    <security-role>
      <role-name>administrator</role-name>
    </security-role>
</application>
```

関連要素

- <module> 要素
- <env-def> 要素
- <property> 要素
- <authorization-domain> 要素
- <security-role> 要素

<authorization-domain> 要素

```
<xsd:element name="authorization-domain" type="xsd:string" minOccurs="0"/>
```

authorization-domain 要素は、有効なユーザー ロールの定義セットを判断するための認証ドメインを指定します。

サンプル

```
<application>
    ...
    <authorization-domain>default</authorization-domain>
    ...
</application>
```

関連要素

親要素

- <application> 要素

子要素

- なし

<connector> 要素

```
<xsd:element name="connector" type="xsd:string"/>
```

オプションの connector 要素では、リソース アダプタのアーカイブ ファイルの URI を、アプリケーション パッケージのトップ レベルへの相対パスで指定します。これは、アプリケーション標準ディスクリプタ内のコンテンツに対応しています。

サンプル

```
<application>
    <module>
        <connector>my-resource-adapter.rar</connector>
    </module>
    ...
</application>
```

関連要素

親要素

- <module> 要素

子要素

- なし

<deployment-role> 要素

```
<xsd:element name="deployment-role" type="xsd:string" minOccurs="0"/>
```

アプリケーションが実行される BAS ロールのロール名。この名前にアプリケーションの role-name がマップされます。

サンプル

```
<application>
    <module>
        <ejb>my-ejb.jar</ejb>
    </module>
    <module>
        <web>
            <web-uri>myweb.war</web-uri>
        </web>
    </module>
    <authorization-domain>default</authorization-domain>
    <security-role>
        <role-name>administrator</role-name>
        <deployment-role>administrator</deployment-role>
    </security-role>
</application>
```

関連要素

親要素

- <security-role> 要素

子要素

- なし

<ejb> 要素

```
<xsd: element name="ejb" type="xsd:string"/>
```

オプションの ejb 要素では、EJB jar アーカイブ ファイルの URI を、アプリケーション パッケージのトップ レベルへの相対パスで指定します。これは、アプリケーション標準ディスクリプタ内のコンテンツに対応しています。

サンプル

```
<application>
    <module>
        <ejb>my-ejb.jar</ejb>
    </module>
    ...
</application>
```

関連要素

親要素

- <module> 要素

子要素

- なし

<env-def> 要素

```
<xsd:element name="env-def" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <application> 要素

子要素

なし

<hosts> 要素

```
<xsd:element name="hosts" type="xsd:string" minOccurs="0"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <module> 要素

子要素

- なし

<java> 要素

```
<xsd:element name="java" type="xsd:string"/>
```

java 要素では、java アプリケーション クライアント モジュールの URI を、アプリケーション パッケージのトップ レベルへの相対パスで指定します。

サンプル

```
<application>
    <module>
        <java>my-javalib.jar</java>
    </module>
</application>
```

関連要素

親要素

- <module> 要素

子要素

- なし

<module> 要素

```
<xsd:element name="module" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:choice>
    <xsd:element name="connector" type="xsd:string"/>
    <xsd:element name="ejb" type="xsd:string"/>
    <xsd:element name="java" type="xsd:string"/>
    <xsd:element name="web">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="web-uri" type="xsd:string"/>
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
   </xsd:choice>
   <xsd:element name="hosts" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この要素は、同一コンテナ タイプ内で実行される 1 つ以上のコンポーネントのコレクションを表し、そのタイプの注釈またはデプロイメント ディスクリプタを伴います。`module` 要素には、サブ要素として `connector`、`ejb`、`java`、`web` のいずれか、および `hosts` サブ要素が必要です。

サンプル

```
<application>
   <module>
      <ejb>my-ejb.jar</ejb>
   </module>
   <module>
      <web>
         <web-uri>myweb.war</web-uri>
      </web>
   </module>
   ...
</application>
```

関連要素

親要素

- <application> 要素

子要素

- <connector> 要素
- <ejb> 要素
- <java> 要素
- <web> 要素
- <hosts> 要素

<property> 要素

```
<xsd:element name="property" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="prop-name" type="xsd:string"/>
   <xsd:element name="prop-type" type="xsd:string" minOccurs="0"/>
   <xsd:element name="prop-value" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この要素は、実行時にアプリケーションに必要なプロパティ値を指定するために使用されます。各 `property` エントリは、対応するサブ要素を使用して、プロパティの名前、型、および値を指定します。

サンプル

```
<application>
    <module>
        <ejb>my-ejb.jar</ejb>
    </module>
    <module>
        <web>
            <web-uri>myweb.war</web-uri>
        </web>
    </module>
    <property>
      <prop-name>vbroker.security.disable</prop-name>
      <prop-type>security</prop-type>
      <prop-value>false</prop-value>
    </property>
</application>
```

関連要素

親要素

- <application> 要素

子要素

- <prop-name> 要素
- <prop-type> 要素
- <prop-value> 要素

<prop-name> 要素

<xsd: element name="prop-name" type="xsd:string"/>

設定するプロパティの名前を指定します。

サンプル

<prop-name>vbroker.security.disable</prop-name>

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-type> 要素

<xsd: element name="prop-type" type="xsd:string" minOccurs="0"/>

設定するプロパティの型を指定します。

サンプル

<prop-type>security</prop-type>

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-value> 要素

<xsd: element name="prop-value" type="xsd:string"/>

設定するプロパティの値を指定します。

サンプル

```
<prop-value>false</prop-value>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<role-name> 要素

<xsd:element name="role-name" type="xsd:string"/>

アプリケーションで使用される security-role のロール名。これは、BAS のデプロイ環境のロールにマップされます。

サンプル

```
<application>
   <module>
      <ejb>my-ejb.jar</ejb>
   </module>
   <module>
      <web>
         <web-uri>myweb.war</web-uri>
      </web>
   </module>
   <authorization-domain>default</authorization-domain>
   <security-role>
      <role-name>administrator</role-name>
      <deployment-role>administrator</deployment-role>
   </security-role>
</application>
```

関連要素

親要素

- <security-role> 要素

子要素

なし

<security-role> 要素

<xsd:element name="security-role" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>

```
  <xsd:sequence>
   <xsd:element name="role-name" type="xsd:string"/>
   <xsd:element name="deployment-role" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

アプリケーションのロール（application.xml にある）を Borland AppServer の deployment-role にマップします。

サンプル

```
<security-role>
      <role-name>administrator</role-name>
      <deployment-role>administrator</deployment-role>
</security-role>
```

関連要素

親要素

- <application> 要素

子要素

- <role-name> 要素
- <deployment-role> 要素

<web> 要素

```
<xsd:element name="web">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="web-uri" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この Web 要素は、アプリケーション内の Web アプリケーション モジュールの web-uri を定義します。

サンプル

```
<application>
    ...
    <module>
       <web>
          <web-uri>myweb.war</web-uri>
       </web>
    </module>
    ...
</application>
```

関連要素

親要素

- <module> 要素

子要素

- <web-uri> 要素

<web-uri> 要素

```
<xsd:element name="web-uri" type="xsd:string"/>
```

オプションの web-uri 要素では、Web アプリケーション ファイルの URI を、そのアプリケーション パッケージのトップ レベルへの相対パスで指定します。

サンプル

```
<application>
    ...
    <module>
        <web>
            <web-uri>myweb.war</web-uri>
        </web>
    </module>
    ...
</application>
```

関連要素

親要素

- <web> 要素

子要素

- なし

第 3 章

アプリケーション クライアント モジュール：application-client-borland.xml

XSD: application-client_1_4-borland.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.borland.com/devsupport/appserver/
xml/ns/j2ee" xmlns="http://www.w3.org/2001/XMLSchema" xmlns:borl="http://
www.borland.com/devsupport/appserver/xml/ns/j2ee" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.4">
 <!-- Start definition of ComplexTypes -->
 <xsd:complexType name="ejb-refType">
  <xsd:sequence>
   <xsd:element name="ejb-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="resource-refType">
  <xsd:sequence>
   <xsd:element name="res-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="resource-env-refType">
  <xsd:sequence>
   <xsd:element name="resource-env-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="message-destination-refType">
  <xsd:sequence>
   <xsd:element name="message-destination-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="message-destinationType">
  <xsd:sequence>
   <xsd:element name="message-destination-name" type="xsd:string"/>
```

```
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:element name="application-client">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="ejb-ref" type="borl:ejb-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <xsd:element name="resource-ref" type="borl:resource-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <xsd:element name="resource-env-ref" type="borl:resource-env-refType"
minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="message-destination-ref" type="borl:message-destination-
refType" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="message-destination" type="borl:message-
destinationType" minOccurs="0" maxOccurs="unbounded"/>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

<application-client> 要素

```
<xsd:element name="application-client">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="ejb-ref" type="borl:ejb-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <xsd:element name="resource-ref" type="borl:resource-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <xsd:element name="resource-env-ref" type="borl:resource-env-refType"
minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="message-destination-ref" type="borl:message-destination-
refType" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="message-destination" type="borl:message-
destinationType" minOccurs="0" maxOccurs="unbounded"/>
   </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

この要素は `application-client-borland.xml` のルート ノードで、VisiClient コンテナで実行されるアプリケーション クライアントに必要な、ランタイム情報を定義します。このノードには、1 つ以上の `ejb-ref`、`resource-ref`、`resource-env-ref`、`message-destination-ref`、および `message-destination` サブ要素が含まれています。

サンプル

```
<application-client>
    <ejb-ref>
        <ejb-ref-name>ejb/Sort</ejb-ref-name>
        <jndi-name>sort</jndi-name>
    </ejb-ref>
    <resource-ref>
        <res-ref-name>jdbc/CheckingDataSource</res-ref-name>
        <jndi-name>datasources/OracleDataSource</jndi-name>
    </resource-ref>
</application-client>
```

関連要素

- <ejb-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination-ref> 要素
- <message-destination> 要素

<ejb-ref> 要素

```
<xsd: element name="ejb-ref" type="borl:ejb-refType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="ejb-refType">
  <xsd:sequence>
   <xsd:element name="ejb-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
```

この要素は、クライアントによって使用される EJB リファレンスを定義するために使用されます。各 EJB リファレンスには、クライアント アプリケーションによって使用される `ejb-ref-name` と、それに関連付けられた `jndi-name` （該当する場合）が含まれています。

サンプル

```
<ejb-ref>
    <ejb-ref-name>ejb/Sort</ejb-ref-name>
    <jndi-name>sort</jndi-name>
</ejb-ref>
```

関連要素

親要素

- なし

子要素

- <ejb-ref-name> 要素
- <jndi-name> 要素

重要 最新のドキュメントについては、`www.borland.com/techpubs/bes` を参照してください。

<ejb-ref-name> 要素

```
<xsd:element name="ejb-ref-name" type="xsd:string"/>
```

この要素は、クライアント アプリケーションによってリソース リファレンスとして使用される EJB の名前を提供します。

サンプル

```
<ejb-ref-name>ejb/Sort</ejb-ref-name>
```

関連要素

親要素

- <ejb-ref> 要素

子要素

- なし

<jndi-name> 要素

```
<xsd:element ref="borl:jndi-name" minOccurs="0"/>

<xsd:element name="jndi-name" type="xsd:string"/>
```

この要素は、JNDI サービス下の場所を表します。これにより、JDBC データソース、JMS 接続ファクトリ、JMS の送信先など、クライアント アプリケーションによって参照されるリソースの場所が解決されます。

サンプル

```
<jndi-name>jms/Tibco/Queue1</jndi-name>
```

関連要素

親要素

- <ejb-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination> 要素
- <message-destination-ref> 要素

子要素

- なし

<message-destination> 要素

```
<xsd:element name="message-destination" type="borl:message-destinationType"
minOccurs="0" maxOccurs="unbounded"/>

<xsd:complexType name="message-destinationType">
  <xsd:sequence>
   <xsd:element name="message-destination-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

この要素は、メッセージの送信先（JMS のキューやトピックなど）を定義するために使用されます。この要素は、アプリケーション クライアント内に 1 つ以上ある、`message-destination-ref` 要素の `message-destination-link` と対応していなければなりません。各メッセージの送信先には、`message-destination-link` 値に一致する `message-destination-name` と、関連付けられている `jndi-name` が含まれます。

サンプル

```
<application-client>
   ...
   <message-destination>
     <message-destination-name>myAppQueue</message-destination-name>
     <jndi-name>jms/queues/TibcoQueue</jndi-name>
   </message-destination>
   ...
</application-client>
```

関連要素

親要素

- <application-client> 要素

子要素

- <message-destination-name> 要素
- <jndi-name> 要素

<message-destination-name> 要素

```
<xsd:element name="message-destination-name" type="xsd:string"/>
```

この要素は、JMS キューやトピックなど、対象となるメッセージの送信先に割り当てられる論理名を指定します。この名前は、標準アプリケーション クライアント ディスクリプタ内で `message-destination-ref` の `message-destination-link` 要素によって指定されるターゲットの送信先を識別し、アプリケーション内のメッセージ フローを示します。

サンプル

標準アプリケーション クライアント ディスクリプタ application-client.xml:

```
<application-client>
 ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
     <message-destination-type>javax.jms.Queue</message-destination-type>
     <message-destination-usage>Consumes</message-destination-usage>
     <message-destination-link>myAppQueue</message-destination-link>
   </message-destination-ref>
 ...
   <message-destination>
     <message-destination-name>myAppQueue</message-destination-name>
   </message-destination>
</application-client>
```

Borland アプリケーション クライアント ディスクリプタ application-client-borland.xml:

```
<application-client>
 ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
     <jndi-name>jms/queues/Queue1</message-destination-type>
   </message-destination-ref>
 ...
   <message-destination>
     <message-destination-name>myAppQueue</message-destination-name>
     <jndi-name>jms/queues/TibcoQueue</jndi-name>
   </message-destination>
</application-client>
```

`message-destination-link` を介して、このアプリケーションが、名前が **jms/StockQueue** の `message-destination-ref` に対して JNDI ルックアップを実行した場合、`message-destination` の `jndi-name` **jms/queues/TibcoQueue** が使用され、`jndi-name` **jms/queues/Queue1** は使用されてない点に注意してください。

関連要素

親要素

- <message-destination> 要素

子要素

- なし

<message-destination-ref> 要素

```
<xsd:element name="message-destination-ref" type="borl:message-destination-
refType" minOccurs="0" maxOccurs="unbounded"/>

<xsd:complexType name="message-destination-refType">
  <xsd:sequence>
   <xsd:element name="message-destination-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

この要素は、JMS キューやトピックなど、メッセージの送信先リファレンスを定義するために使用されます。各 EJB リファレンスには、アプリケーションによって使用される `message-destination-ref-name` と、それに関連付けられた `jndi-name` が含まれています。

サンプル

```
<application-client>
   ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
      <jndi-name>jms/queues/Queue1</message-destination-type>
   </message-destination-ref>
   ...
</application-client>
```

関連要素

親要素

- <application-client> 要素

子要素

- <message-destination-ref-name> 要素
- <jndi-name> 要素

<message-destination-ref-name> 要素

```
<xsd:element name="message-destination-ref-name" type="xsd:string"/>
```

この要素は、JMS キューやトピックなど、メッセージの送信先リファレンスにアクセスするために、クライアント アプリケーションが使用する論理名を指定します。この名前は、アプリケーション コンポーネントの java:comp/env への JNDI 相対名です。

サンプル

```
<application-client>
   ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
      <jndi-name>jms/queues/Queue1</message-destination-type>
   </message-destination-ref>
   ...
</application-client>
```

関連要素

親要素

- <message-destination-ref> 要素

子要素

- なし

<resource-env-ref-name> 要素

```
<xsd:element name="resource-env-ref-name" type="xsd:string"/>
```

この要素は、クライアント アプリケーションがリソース環境リファレンスにアクセスするために使用する名前を提供します。これは、標準デプロイメント ディスクリプタからリソースの環境リファレンスを一意に識別します。

サンプル

```
<application-client>
  ...
   <resource-env-ref>
     <resource-env-ref-name>jms/StockQueue</resource-env-ref-name>
     <jndi-name>jms/Queue1</jndi-name>
   </resource-env-ref>
...
</application-client>
```

関連要素

親要素

- <resource-env-ref> 要素

子要素

- なし

<res-ref-name> 要素

<xsd: element name="res-ref-name" type="xsd:string"/>

この要素は、クライアント アプリケーションがリソース リファレンスにアクセスするために使用する名前を提供します。これは、標準デプロイメント ディスクリプタからリソースの環境リファレンスを一意に識別します。

サンプル

```
<application-client>
  ...
    <resource-ref>
      <res-ref-name>jdbc/SavingsDataSource</res-ref-name>
      <jndi-name>jdbc/datasources/Oracle</jndi-name>
    </resource-ref>
...
</application-client>
```

関連要素

親要素

- <resource-ref> 要素

子要素

- なし

<resource-env-ref> 要素

```
<element name="resource-env-ref" type="borl:resource-env-refType"
minOccurs="0" maxOccurs="unbounded"/>

<complexType name="resource-env-refType">
  <sequence>
    <element name="resource-env-ref-name" type="xsd:string"/>
    <element ref="borl:jndi-name"/>
  </sequence>
</complexType>
```

この要素は、クライアントによって使用されるリソース環境リファレンスを定義するために使用されます。各リソース環境リファレンスには、クライアント アプリケーションによって使用される `resource-env-ref-name` と、それに関連付けられた `jndi-name` （該当する場合）が含まれています。`resource-env-ref-name` 要素は、標準デプロイメント ディスクリプタからリソースの環境リファレンスを一意に識別します。

サンプル

```
<application-client>
   ...
   <resource-env-ref>
      <resource-env-ref-name>jms/StockQueue</resource-env-ref-name>
      <jndi-name>jms/Queue1</jndi-name>
   </resource-env-ref>
   ...
</application-client>
```

関連要素

親要素

- <application-client> 要素

子要素

- <resource-env-ref-name> 要素
- <jndi-name> 要素

<resource-ref> 要素

```
<xsd:element name="resource-ref" type="borl:resource-refType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="resource-refType">
  <xsd:sequence>
   <xsd:element name="res-ref-name" type="xsd:string"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

この要素は、クライアントによって使用されるリソース リファレンスを定義するために使用されます。各リソース リファレンスには、クライアント アプリケーションによって使用される `res-ref-name` と、それに関連付けられた `jndi-name`（該当する場合）が含まれています。`res-ref-name` 要素は、標準デプロイメント ディスクリプタからリソースのリファレンスを一意に識別します。

サンプル

```
<application-client>
  ...
    <resource-ref>
      <res-ref-name>jdbc/SavingsDataSource</res-ref-name>
      <jndi-name>jdbc/datasources/Oracle</jndi-name>
    </resource-ref>
...
</application-client>
```

関連要素

親要素

- <application-client> 要素

子要素

- <res-ref-name> 要素
- <jndi-name> 要素

第 4 章

コネクタ モジュール : ra-borland.xml

XSD: connector_1_5-borland.xsd

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema targetNamespace="http://
support.borland.com/appserver/xml/ns/j2ee" xmlns="http://www.w3.org/2001/
XMLSchema" xmlns:borl="http://support.borland.com/appserver/xml/ns/j2ee"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.5">
 <xsd:complexType name="resourceadapterType">
  <xsd:sequence>
   <xsd:element name="instance-name" type="xsd:string"/>
   <xsd:element name="outbound-resourceadapter" type="borl:outbound-
resourceadapterType" minOccurs="0">
    <xsd:unique name="connectionfactory-interface-uniqueness">
     <xsd:selector xpath="borl:connection-definition"/>
     <xsd:field xpath="borl:connectionfactory-interface"/>
    </xsd:unique>
   </xsd:element>
   <xsd:element name="ra-link-ref" type="xsd:string" minOccurs="0"/>
   <xsd:element name="ra-libraries" type="xsd:string" minOccurs="0"/>
   <xsd:element name="security-map" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="description" type="xsd:string" minOccurs="0"/>
      <xsd:element name="user-role" type="xsd:string" maxOccurs="unbounded"/>
      <xsd:choice>
       <xsd:element name="use-caller-identity"/>
       <xsd:element name="run-as">
        <xsd:complexType>
         <xsd:sequence>
          <xsd:element name="description" type="xsd:string" minOccurs="0"/>
          <xsd:element name="role-name" type="xsd:string"/>
         </xsd:sequence>
        </xsd:complexType>
       </xsd:element>
      </xsd:choice>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
```

```
   <xsd:element name="authorization-domain" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID"/>
 </xsd:complexType>
 <xsd:complexType name="outbound-resourceadapterType">
  <xsd:sequence>
   <xsd:element name="connection-definition" type="borl:connection-definitionType"
maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="connection-definitionType">
  <xsd:sequence>
   <xsd:element name="connectionfactory-interface" type="xsd:string"/>
   <xsd:element name="factory-name" type="xsd:string"/>
   <xsd:element name="factory-description" type="xsd:string" minOccurs="0"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
   <xsd:element name="pool-parameters" minOccurs="0">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="initial-capacity" type="xsd:string" minOccurs="0"/>
      <xsd:element name="maximum-capacity" type="xsd:string" minOccurs="0"/>
      <xsd:element name="capacity-delta" type="xsd:string" minOccurs="0"/>
      <xsd:element name="cleanup-enabled" type="xsd:string" minOccurs="0"/>
      <xsd:element name="cleanup-delta" type="xsd:string" minOccurs="0"/>
      <xsd:element name="busy-timeout" type="xsd:string" minOccurs="0"/>
      <xsd:element name="idle-timeout" type="xsd:string" minOccurs="0"/>
      <xsd:element name="wait-timeout" type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:sequence minOccurs="0">
    <xsd:element name="logging-enabled" type="xsd:string"/>
    <xsd:element name="log-file-name" type="xsd:string"/>
   </xsd:sequence>
   <xsd:element name="property" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="prop-name" type="xsd:string"/>
      <xsd:element name="prop-type" type="xsd:string"/>
      <xsd:element name="prop-value" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:element name="connector">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="resourceadapter" type="borl:resourceadapterType"/>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
</xsd:schema>
```

<authorization-domain> 要素

```
<xsd:element name="authorization-domain" type="xsd:string" minOccurs="0"/>
```

authorization-domain 要素は、有効なユーザー ロールの定義セットを判断するための認証ドメインを指定します。

サンプル

```
<connector>
    <resourceadapter>
        ...
        <authorization-domain>default</authorization-domain>
    </resourceadapter>
    ...
</connector>
```

関連要素

親要素

- <resourceadapter> 要素

子要素

- なし

<busy-timeout> 要素

```
<xsd:element name="busy-timeout" type="xsd:string" minOccurs="0"/>
```

ビジー接続が解放されるまで待つ時間を秒単位で指定します。指定しない場合は、デフォルト値の 600 が使用されます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
               <factory-name>mailConnectionFactory</factory-name>
               <jndi-name>serial://jca/mail/cf</jndi-name>
               <pool-parameters>
                <idle-timeout>500</idle-timeout>
                <busy-timeout>300</busy-timeout>
               </pool-parameters>
            </connection-definition>
         </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<capacity-delta> 要素

```
<xsd:element name="capacity-delta" type="xsd:string" minOccurs="0"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<cleanup-delta> 要素

```
<xsd:element name="cleanup-delta" type="xsd:string" minOccurs="0"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<cleanup-enabled> 要素

```
<xsd:element name="cleanup-enabled" type="xsd:string" minOccurs="0"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<connection-definition> 要素

```
<xsd:element name="connection-definition" type="borl:connection-definitionType"
maxOccurs="unbounded"/>

<xsd:complexType name="connection-definitionType">
  <xsd:sequence>
   <xsd:element name="connectionfactory-interface" type="xsd:string"/>
   <xsd:element name="factory-name" type="xsd:string"/>
   <xsd:element name="factory-description" type="xsd:string" minOccurs="0"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
   <xsd:element name="pool-parameters" minOccurs="0">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="initial-capacity" type="xsd:string" minOccurs="0"/>
      <xsd:element name="maximum-capacity" type="xsd:string" minOccurs="0"/>
      <xsd:element name="capacity-delta" type="xsd:string" minOccurs="0"/>
      <xsd:element name="cleanup-enabled" type="xsd:string" minOccurs="0"/>
      <xsd:element name="cleanup-delta" type="xsd:string" minOccurs="0"/>
      <xsd:element name="busy-timeout" type="xsd:string" minOccurs="0"/>
      <xsd:element name="idle-timeout" type="xsd:string" minOccurs="0"/>
      <xsd:element name="wait-timeout" type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:sequence minOccurs="0">
    <xsd:element name="logging-enabled" type="xsd:string"/>
    <xsd:element name="log-file-name" type="xsd:string"/>
   </xsd:sequence>
   <xsd:element name="property" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="prop-name" type="xsd:string"/>
      <xsd:element name="prop-type" type="xsd:string"/>
      <xsd:element name="prop-value" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
```

connection-definition 要素は、デプロイされた送信リソース アダプタの Borland 固有の接続情報を定義します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
           <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
               <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <ra-libraries>native/lib</ra-libraries>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <outbound-resourceadapter> 要素

子要素

- <connectionfactory-interface> 要素
- <factory-name> 要素
- <factory-description> 要素
- <jndi-name> 要素
- <pool-parameters > 要素
- <logging-enabled> 要素
- <log-file-name> 要素
- <property> 要素

<connectionfactory-interface> 要素

```
<xsd:element name="connectionfactory-interface" type="xsd:string"/>

<xsd:field xpath="borl:connectionfactory-interface"/>
```

connectionfactory-interface 要素は、親の connection-definition がサポートする ConnectionFactory インターフェイスの完全修飾名を指定します。この修飾名は、対応する connection-definition を標準ディスクリプタから一意に識別します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- なし

<connector> 要素

```
<xsd:element name="connector">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="resourceadapter" type="borl:resourceadapterType"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

connector 要素は、リソース アダプタの Borland のデプロイメント ディスクリプタのルート要素です。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
               <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <ra-libraries>native/lib</ra-libraries>
   </resourceadapter>
</connector>
```

関連要素

親要素

- なし

子要素

- <resourceadapter> 要素

<description> 要素

```
<xsd:element name="description" type="xsd:string" minOccurs="0"/>
```

セキュリティ マップまたは run-as ロール要素の説明を指定します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <wait-timeout>70</idle-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <security-map>
         <description>Mapping of user role</description>
         <user-role>Administrator</user-role>
         <run-as>
            <description>Target role mapped from user role </description>
            <role-name>admin</role-name>
      </security-map>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <security-map> 要素
- <run-as> 要素

子要素

- なし

<factory-description> 要素

```
<xsd:element name="factory-description" type="xsd:string" minOccurs="0"/>
```

factory-description 要素は、リソース アダプタによってサポートされる ConnectionFactory の説明を取得します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <factory-description>Details of configuration required to connect to
outbound Mail Adapter</factory-description>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <ra-libraries>native/lib</ra-libraries>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- なし

<factory-name> 要素

```
<xsd:element name="factory-name" type="xsd:string"/>
```

factory-name 要素は、AppServer パーティションにデプロイされたすべてのリソース アダプタの間で接続ファクトリを一意に識別します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <factory-description>Details of configuration required to connect to
outbound Mail Adapter</factory-description>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- なし

<idle-timeout> 要素

```
<xsd:element name="idle-timeout" type="xsd:string" minOccurs="0"/>
```

親の接続定義要素に関連付けられた接続プールから取得された接続は、idle-timeout 要素によって指定されたタイムアウト値よりも長い時間アイドル状態が続いた場合は、終了する必要があります。アイドル接続に対して、60 秒ごとに期限切れが確認されます。idle-timeout の値の単位は秒数です。デフォルトは 600 秒です。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
              <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<initial-capacity> 要素

```
<xsd:element name="initial-capacity" type="xsd:string" minOccurs="0"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<instance-name> 要素

```
<xsd:element name="instance-name" type="xsd:string"/>
```

instance-name に指定されたこの値は、Borland AppServer パーティションにデプロイされた RAR の間で一意である必要があります。これは、デプロイされたリソース アダプタを VisiConnect サービスが一意に識別するために使用されます。これは、リソース アダプタが着信をサポートしている場合に、受信メッセージを受け取る予定のリソース アダプタを識別するために、エンドポイント MDB が ejb-borland.xml ディスクリプタ内で使用する名前です。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
            </pool-parameters>
            <logging-enabled>true</logging-enabled>
            <log-file-name>mail_adapter.log</log-file-name>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <resourceadapter> 要素

子要素

- なし

<jndi-name> 要素

```
<xsd:element name="jndi-name" type="xsd:string"/>
```

connection-definition の jndi-name 要素は、JNDI 下の接続ファクトリを識別します。これは、このパーティションにデプロイされたすべてのリソース アダプタの間で一意である必要があります。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
       </outbound-resourceadapter>
      <ra-libraries>native/lib</ra-librarie>
    </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- なし

<log-file-name> 要素

```
<xsd:element name="log-file-name" type="xsd:string"/>
```

logging-enabled 要素は、ManagedConnectionFactory または ManagedConnection に対してログライターが設定されているかどうかを示します。この要素が true に設定された場合は、ManagedConnectionFactory または ManagedConnection から生成された出力が log-file-name 要素で指定されたファイルに送信されます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
            </pool-parameters>
            <logging-enabled>true</logging-enabled>
            <log-file-name>mail_adapter.log</log-file-name>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- なし

<logging-enabled> 要素

```
<xsd:element name="logging-enabled" type="xsd:string"/>
```

logging-enabled 要素は、ManagedConnectionFactory または ManagedConnection に対してログライターが設定されているかどうかを示します。この要素が true に設定された場合は、ManagedConnectionFactory または ManagedConnection から生成された出力が log-file-name 要素で指定されたファイルに送信されます。指定しない場合は、デフォルト値の false が使用されます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
            </pool-parameters>
            <logging-enabled>true</logging-enabled>
            <log-file-name>mail_adapter.log</log-file-name>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- なし

<maximum-capacity> 要素

```
<xsd:element name="maximum-capacity" type="xsd:string" minOccurs="0"/>
```

maximum-capacity 要素は、AppServer パーティションのコネクタ サーバーが取得できる管理接続の最大数を指定します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <maximum-capacity>10</capacity-delta>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

<outbound-resourceadapter> 要素

```
<xsd:element name="outbound-resourceadapter" type="borl:outbound-
resourceadapterType" minOccurs="0">
    <xsd:unique name="connectionfactory-interface-uniqueness">
        <xsd:selector xpath="borl:connection-definition"/>
        <xsd:field xpath="borl:connectionfactory-interface"/>
    </xsd:unique>
</xsd:element>

<xsd:complexType name="outbound-resourceadapterType">
    <xsd:sequence>
        <xsd:element name="connection-definition" type="borl:connection-
definitionType" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
```

outbound-resourceadapter 要素は、送信リソース アダプタの追加設定情報を指定します。

サンプル

```
<connector>
  <resourceadapter>
    <instance-name>mail_adapter</instance-name>
    <outbound-resourceadapter>
      <connection-definition>
        <connectionfactory-interface>
                com.borland.enterprise.ra.mail.api.MailConnectionFactory<
                /connectionfactory-interface>
        <factory-name>mailConnectionFactory</factory-name>
        <jndi-name>serial://jca/mail/cf</jndi-name>
        <pool-parameters>
          <idle-timeout>500</idle-timeout>
          <busy-timeout>300</busy-timeout>
          <wait-timeout>70</wait-timeout>
        </pool-parameters>
      </connection-definition>
    </outbound-resourceadapter>
  </resourceadapter>
</connector>
```

関連要素

親要素

- なし

子要素

- <connection-definition> 要素

<pool-parameters> 要素

```
<xsd:element name="pool-parameters" minOccurs="0">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="initial-capacity" type="xsd:string" minOccurs="0"/>
   <xsd:element name="maximum-capacity" type="xsd:string" minOccurs="0"/>
   <xsd:element name="capacity-delta" type="xsd:string" minOccurs="0"/>
   <xsd:element name="cleanup-enabled" type="xsd:string" minOccurs="0"/>
   <xsd:element name="cleanup-delta" type="xsd:string" minOccurs="0"/>
   <xsd:element name="busy-timeout" type="xsd:string" minOccurs="0"/>
   <xsd:element name="idle-timeout" type="xsd:string" minOccurs="0"/>
   <xsd:element name="wait-timeout" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

pool-parameters 要素は、関連付けられている親接続の定義に対して、接続プールに固有のパラメータを提供します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
              <idle-timeout>500</idle-timeout>
              <busy-timeout>300</busy-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connection-definition> 要素

子要素

- <initial-capacity> 要素
- <maximum-capacity> 要素
- <capacity-delta> 要素
- <cleanup-enabled> 要素
- <cleanup-delta> 要素
- <busy-timeout> 要素
- <idle-timeout> 要素
- <wait-timeout> 要素

<property> 要素

```
<xsd:element name="property" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="prop-name" type="xsd:string"/>
   <xsd:element name="prop-type" type="xsd:string"/>
   <xsd:element name="prop-value" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この要素は使用されなくなりました。

関連要素

親要素

- <connection-definition> 要素

子要素

- <prop-name> 要素
- <prop-type> 要素
- <prop-value> 要素

<prop-name> 要素

```
<xsd:element name="prop-name" type="xsd:string"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-type> 要素

```
<xsd:element name="prop-type" type="xsd:string"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-value> 要素

```
<xsd:element name="prop-value" type="xsd:string"/>
```

この要素は使用されなくなりました。

関連要素

親要素

- <property> 要素

子要素

- なし

<ra-libraries> 要素

```
<xsd:element name="ra-libraries" type="xsd:string" minOccurs="0"/>
```

ra-libraries 要素は、このリソース アダプタ デプロイメント内に存在するネイティブ ライブラリのために使用されるディレクトリの場所を指定します。デプロイメント プロセスの一部として、検知されたネイティブ ライブラリは、すべて指定された場所にコピーされます。

必要なプラットフォーム アクションを実行して、実行時にこれらのライブラリが検出されるようにするのは、管理者の役割です。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <ra-libraries>native/lib</ra-libraries>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <resourceadapter> 要素

子要素

- なし

<ra-link-ref> 要素

<xsd: element name="ra-link-ref" type="xsd:string" minOccurs="0"/>

ra-link-ref 要素を使用すると、デプロイ済み複数の接続ファクトリを、デプロイ済みの 1 つのリソース アダプタに論理的に関連付けることができます。このオプションの `ra-link-ref` 要素を、デプロイ済みの個々の接続ファクトリを識別できる値と共に指定すると、その新たにデプロイされる接続ファクトリは、参照されている接続ファクトリと共にすでにデプロイされているリソース アダプタを、共有するようになります。

また、リファレンス先の接続ファクトリのデプロイメントで定義されている値は、特に指定されない限り、この新しくデプロイされた接続ファクトリによって継承されます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
       </outbound-resourceadapter>
      <ra-link-ref>jmsConnectionFactory</ra-link-ref>
    </resourceadapter>
</connector>
```

関連要素

親要素

- <resourceadapter> 要素

子要素

- なし

<resourceadapter> 要素

```
<xsd:element name="resourceadapter" type="borl:resourceadapterType"/>

<xsd:complexType name="resourceadapterType">
 <xsd:sequence>
  <xsd:element name="instance-name" type="xsd:string"/>
  <xsd:element name="outbound-resourceadapter" type="borl:outbound-
resourceadapterType" minOccurs="0">
   <xsd:unique name="connectionfactory-interface-uniqueness">
    <xsd:selector xpath="borl:connection-definition"/>
    <xsd:field xpath="borl:connectionfactory-interface"/>
   </xsd:unique>
  </xsd:element>
  <xsd:element name="ra-link-ref" type="xsd:string" minOccurs="0"/>
  <xsd:element name="ra-libraries" type="xsd:string" minOccurs="0"/>
  <xsd:element name="security-map" minOccurs="0" maxOccurs="unbounded">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="description" type="xsd:string" minOccurs="0"/>
     <xsd:element name="user-role" type="xsd:string" maxOccurs="unbounded"/>
     <xsd:choice>
      <xsd:element name="use-caller-identity"/>
      <xsd:element name="run-as">
       <xsd:complexType>
        <xsd:sequence>
         <xsd:element name="description" type="xsd:string" minOccurs="0"/>
         <xsd:element name="role-name" type="xsd:string"/>
        </xsd:sequence>
       </xsd:complexType>
      </xsd:element>
     </xsd:choice>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="authorization-domain" type="xsd:string" minOccurs="0"/>
 </xsd:sequence>
 <xsd:attribute name="id" type="xsd:ID"/>
</xsd:complexType>
```

resourceadapter 要素は、リソース アダプタの追加情報を指定します。この情報は、リソース アダプタのデプロイメント時に Borland AppServer によって使用されます。リソース アダプタを一意に識別する名前と、outbound-resourceadapter 設定情報が含まれます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <connector> 要素

子要素

- <instance-name> 要素
- <outbound-resourceadapter> 要素
- <ra-link-ref> 要素
- <ra-libraries> 要素
- <security-map> 要素
- <authorization-domain> 要素

<role-name> 要素

```
<xsd: element name="role-name" type="xsd:string"/>
```

role-name 要素には、セキュリティ ロールの名前を指定します。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <security-map>
         <description>Mapping of user role</description>
         <user-role>Administrator</user-role>
         <run-as>
            <role-name>admin</role-name>
         </run-as>
      </security-map>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <run-as> 要素

子要素

- なし

<run-as> 要素

```
<xsd:element name="run-as">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="description" type="xsd:string" minOccurs="0"/>
   <xsd:element name="role-name" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

run-as 要素は、エンタープライズ Bean を実行するために使用する run-as ID を指定します。この要素には、オプションの説明とセキュリティ ロールの名前を含めます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <security-map>
         <description>Mapping of user role</description>
         <user-role>Administrator</user-role>
         <run-as>
            <role-name>admin</role-name>
         </run-as>
      </security-map>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <security-map> 要素

子要素

- <description> 要素
- <role-name> 要素

<security-map> 要素

```
<xsd:element name="security-map" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="description" type="xsd:string" minOccurs="0"/>
   <xsd:element name="user-role" type="xsd:string" maxOccurs="unbounded"/>
   <xsd:choice>
    <xsd:element name="use-caller-identity"/>
    <xsd:element name="run-as">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="description" type="xsd:string" minOccurs="0"/>
       <xsd:element name="role-name" type="xsd:string"/>
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
   </xsd:choice>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

`security-map` 要素は、エンタープライズ Bean のメソッドを実行する際に呼び出し元のセキュリティ ID を使用するかどうか、または特定の run-as ID を使用するかどうかを指定します。この要素には、使用するセキュリティ ID のオプションの説明と指定を指定します。

各 `security-map` 要素は、run-as 要素を使用した、リソース アダプタ /EIS 認証処理のための適切なリソース ロールを定義するメカニズムを提供します。

この要素では、管理接続と接続ハンドルを割り当てる際に使用する定義済みのユーザー ロールのセットと、それに対応する run-as ロール（EIS ID を表す）を指定できます。

マップを使用して、デフォルトのリソース run-as ロールを接続ファクトリに対して定義できます。user-role 値 * と、それに対応する run-as ロールを指定すると、ロールがマップ内の他の場所で一致しない場合は、定義済みの run-as が使用されます。

この要素は省略可能ですが、コンテナ管理のサインオンがリソース アダプタによってサポートされ、いずれかのクライアントで使用されている場合は、なんらかの形で指定する必要があります。また、管理接続を持つ接続プールは、定義済みのデフォルトの run-as ロールが指定されている場合、そのデフォルトを使ってデプロイメント時に生成が試みられます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <security-map>
        <user-role>Administrator</user-role>
        <use-caller-identity/>
      </security-map>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <resourceadapter> 要素

子要素

- <description> 要素
- <user-role> 要素
- <use-caller-identity> 要素
- <run-as> 要素

<use-caller-identity> 要素

<xsd: element name="use-caller-identity">

use-caller-identity 要素は、リソース アダプタのメソッドを実行するためのセキュリティ ID として使用する呼び出し元のセキュリティ ID を指定します。これは空の要素です。使用しない場合は、かわりに run-as 要素を指定する必要があります。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <security-map>
         <description>Mapping of user role</description>
         <user-role>Administrator</user-role>
         <use-caller-identity/>
      </security-map>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <security-map> 要素

子要素

- なし

<user-role> 要素

<xsd: element name="user-role" type="xsd:string" maxOccurs="unbounded"/>

`user-role` 要素には、リソースとの通信のために使用される、1 つ以上のロール名を指定します。これは、セキュリティ ID としてそのまま使用されるために定義されるか、もしくは、適切なリソース ロール run-as ID にマップされます。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
               <factory-name>mailConnectionFactory</factory-name>
               <jndi-name>serial://jca/mail/cf</jndi-name>
               <pool-parameters>
                  <wait-timeout>70</wait-timeout>
               </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
      <security-map>
         <description>Mapping of user role</description>
         <user-role>Administrator</user-role>
         <run-as>
            <description>Target role mapped from user role </description>
            <role-name>admin</role-name>
         </run-as>
      </security-map>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <security-map> 要素

子要素

- なし

<wait-timeout> 要素

```
<xsd:element name="wait-timeout" type="xsd:string" minOccurs="0"/>
```

wait-timeout 要素は、親要素である connection-definition 要素に関連付けられた接続プールから、接続が解放されるまで待機する秒数を指定します。デフォルトは 30 秒です。

サンプル

```
<connector>
   <resourceadapter>
      <instance-name>mail_adapter</instance-name>
      <outbound-resourceadapter>
         <connection-definition>
            <connectionfactory-
interface>com.borland.enterprise.ra.mail.api.MailConnectionFactory</
connectionfactory-interface>
            <factory-name>mailConnectionFactory</factory-name>
            <jndi-name>serial://jca/mail/cf</jndi-name>
            <pool-parameters>
               <idle-timeout>500</idle-timeout>
               <busy-timeout>300</busy-timeout>
              <wait-timeout>70</wait-timeout>
            </pool-parameters>
         </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>
</connector>
```

関連要素

親要素

- <pool-parameters > 要素

子要素

- なし

第 5 章

EJB モジュール：ejb-borland.xml

XSD: ejb-jar_2_1-borland.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Ray Chapman (Borland Software Corporation) -->
<schema targetNamespace="http://support.borland.com/appserver/xml/ns/j2ee" xmlns:xsd="http://www.w3.org/2001/
XMLSchema" xmlns:borl="http://support.borland.com/appserver/xml/ns/j2ee" xmlns="http://www.w3.org/2001/
XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.1">
 <!-- Start definition of ComplexTypes -->
 <complexType name="admin-objectType">
  <sequence>
   <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
 </complexType>
 <complexType name="ejb-refType">
  <sequence>
   <element name="ejb-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
 <complexType name="ejb-local-refType">
  <sequence>
   <element name="ejb-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
 <complexType name="resource-refType">
  <sequence>
   <element name="res-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string"/>
   <element name="cmp-resource" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
 <complexType name="resource-env-refMdbType">
  <sequence>
   <element name="resource-env-ref-name" type="xsd:string"/>
   <choice>
    <element name="admin-object" type="borl:admin-objectType"/>
    <element name="jndi-name" type="xsd:string"/>
```

```
  </choice>
 </sequence>
</complexType>
<complexType name="resource-env-refType">
 <sequence>
  <element name="resource-env-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="message-destination-refType">
 <sequence>
  <element name="message-destination-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="propertyType">
 <sequence>
  <element name="prop-name" type="xsd:string"/>
  <element name="prop-type" type="xsd:string" minOccurs="0"/>
  <element name="prop-value" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="resource-adapter-refType">
 <sequence>
  <element name="instance-name" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="jms-provider-refType">
 <sequence>
  <element name="message-driven-destination-name" type="xsd:string"/>
  <element name="connection-factory-name" type="xsd:string"/>
  <element name="pool" minOccurs="0">
   <complexType>
    <sequence>
     <element name="max-size" type="xsd:string" minOccurs="0"/>
     <element name="init-size" type="xsd:string" minOccurs="0"/>
     <element name="wait-timeout" type="xsd:string" minOccurs="0"/>
    </sequence>
   </complexType>
  </element>
 </sequence>
</complexType>
<complexType name="table-refType">
 <sequence>
  <element name="left-table">
   <complexType>
    <sequence>
     <element name="table-name" type="xsd:string"/>
     <element name="column-list" type="borl:column-listType"/>
    </sequence>
   </complexType>
  </element>
  <element name="cross-table" minOccurs="0" maxOccurs="unbounded">
   <complexType>
    <sequence>
     <element name="table-name" type="xsd:string"/>
     <element name="column-list" type="borl:column-listType"/>
     <element name="column-list" type="borl:column-listType"/>
    </sequence>
   </complexType>
  </element>
  <element name="right-table">
```

```
   <complexType>
    <sequence>
     <element name="table-name" type="xsd:string"/>
     <element name="column-list" type="borl:column-listType"/>
    </sequence>
   </complexType>
  </element>
 </sequence>
</complexType>
<complexType name="ejb-relationship-roleType">
 <sequence>
  <element name="relationship-role-source">
   <complexType>
    <sequence>
     <element name="ejb-name" type="xsd:string"/>
    </sequence>
   </complexType>
  </element>
  <element name="cmr-field" minOccurs="0">
   <complexType>
    <sequence>
     <element name="cmr-field-name" type="xsd:string"/>
     <element name="table-ref" type="borl:table-refType"/>
     <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
   </complexType>
  </element>
  <element name="cascade-delete-db" minOccurs="0">
   <complexType/>
  </element>
 </sequence>
</complexType>
<complexType name="column-listType">
 <sequence>
  <element name="column-name" type="xsd:string" maxOccurs="unbounded"/>
 </sequence>
</complexType>
<complexType name="ejb-jarType">
 <sequence>
  <element name="enterprise-beans">
   <complexType>
    <choice maxOccurs="unbounded">
     <element name="session">
      <complexType>
       <sequence>
        <element name="ejb-name" type="xsd:string"/>
        <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
        <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
        <element name="timeout" type="xsd:string" minOccurs="0"/>
        <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="resource-ref" type="borl:resource-refType"
minOccurs="0" maxOccurs="unbounded"/>
        <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
        <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
       </sequence>
      </complexType>
     </element>
     <element name="entity">
      <complexType>
```

```
      <sequence>
       <element name="ejb-name" type="xsd:string"/>
       <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
       <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
       <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
       <choice minOccurs="0">
        <element name="cmp-info">
         <complexType>
          <sequence>
           <element name="description" type="xsd:string" minOccurs="0"/>
           <element name="database-map" minOccurs="0">
            <complexType>
             <sequence>
              <element name="table" type="xsd:string" minOccurs="0"/>
              <element name="column-map" minOccurs="0" maxOccurs="unbounded">
               <complexType>
                <sequence>
                 <element name="field-name" type="xsd:string"/>
                 <element name="column-name" type="xsd:string" minOccurs="0"/>
                 <element name="column-type" type="xsd:string" minOccurs="0"/>
                 <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
                </sequence>
               </complexType>
              </element>
             </sequence>
            </complexType>
           </element>
           <element name="finder" minOccurs="0" maxOccurs="unbounded">
            <complexType>
             <sequence>
              <element name="method-signature" type="xsd:string"/>
              <element name="where-clause" type="xsd:string"/>
              <element name="load-state" type="xsd:string" minOccurs="0"/>
             </sequence>
            </complexType>
           </element>
          </sequence>
         </complexType>
        </element>
        <element name="cmp2-info">
         <complexType>
          <sequence>
           <element name="cmp-field" minOccurs="0" maxOccurs="unbounded">
            <complexType>
             <sequence>
              <element name="field-name" type="xsd:string"/>
              <choice>
               <element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
                <complexType>
                 <sequence>
                  <element name="field-name" type="xsd:string"/>
                  <element name="column-name" type="xsd:string"/>
                 </sequence>
                </complexType>
               </element>
               <element name="column-name" type="xsd:string"/>
              </choice>
```

```
          <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
         </sequence>
        </complexType>
       </element>
       <element name="table-name" type="xsd:string"/>
       <element name="table-ref" type="borl:table-refType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
   </choice>
   <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="query" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="query-method">
       <complexType>
        <sequence>
         <element name="method-name" type="xsd:string"/>
         <element name="method-params">
          <complexType>
           <sequence>
            <element name="method-param" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
           </sequence>
          </complexType>
         </element>
        </sequence>
       </complexType>
      </element>
      <element name="user-sql" type="xsd:string" minOccurs="0"/>
      <element name="load-state" type="xsd:string" minOccurs="0"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
<element name="message-driven">
 <complexType>
  <sequence>
   <element name="ejb-name" type="xsd:string"/>
   <element name="message-source">
    <complexType>
     <choice>
      <element name="resource-adapter-ref" type="borl:resource-adapter-refType"/>
      <element name="jms-provider-ref" type="borl:jms-provider-refType"/>
     </choice>
    </complexType>
   </element>
   <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0"
maxOccurs="unbounded"/>
   <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="resource-env-ref" type="borl:resource-env-refMdbType" minOccurs="0"
maxOccurs="unbounded"/>
   <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
   <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
</choice>
</complexType>
```

```
</element>
<element name="datasource-definitions" minOccurs="0">
 <complexType>
  <sequence>
   <element name="datasource" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="jndi-name" type="xsd:string"/>
      <element name="url" type="xsd:string"/>
      <element name="username" type="xsd:string" minOccurs="0"/>
      <element name="password" type="xsd:string" minOccurs="0"/>
      <element name="isolation-level" type="xsd:string" minOccurs="0"/>
      <element name="driver-class-name" type="xsd:string" minOccurs="0"/>
      <element name="jdbc-property" minOccurs="0" maxOccurs="unbounded">
       <complexType>
        <sequence>
         <element name="prop-name" type="xsd:string"/>
         <element name="prop-value" type="xsd:string"/>
        </sequence>
       </complexType>
      </element>
      <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
<element name="table-properties" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="table-name" type="xsd:string"/>
   <element name="column-properties" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="column-name" type="xsd:string"/>
      <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
   <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
<element name="relationships" minOccurs="0">
 <complexType>
  <sequence>
   <element name="ejb-relation" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="ejb-relationship-role" type="borl:ejb-relationship-roleType"/>
      <element name="ejb-relationship-role" type="borl:ejb-relationship-roleType"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
<element name="authorization-domain" type="xsd:string" minOccurs="0"/>
<element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
<element name="assembly-descriptor" minOccurs="0">
 <complexType>
```

```
    <sequence>
     <element name="security-role" minOccurs="0" maxOccurs="unbounded">
      <complexType>
       <sequence>
        <element name="role-name" type="xsd:string"/>
        <element name="deployment-role" type="xsd:string" minOccurs="0"/>
       </sequence>
      </complexType>
     </element>
     <element name="message-destination" minOccurs="0" maxOccurs="unbounded">
      <complexType>
       <sequence>
        <element name="message-destination-name" type="xsd:string"/>
        <element name="jndi-name" type="xsd:string"/>
       </sequence>
      </complexType>
     </element>
    </sequence>
   </complexType>
  </element>
 </sequence>
</complexType>
<!-- End definition of ComplexTypes -->
<!-- Definition of XML instance content -->
<element name="ejb-jar" type="borl:ejb-jarType"/>
</schema>
```

<admin-object> 要素

```
<element name="admin-object" type="borl:admin-objectType"/>

<complexType name="admin-objectType">
   <sequence>
      <element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
   </sequence>
</complexType>
```

アプリケーションは、この要素を使用することにより、リソース アダプタが提供する 1 つ以上の JavaBean 管理オブジェクトを解決できます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <resource-adapter-ref>
               <instance-name>mailAdapter</instance-name>
            </resource-adapter-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>resource/adminobject1</resource-env-ref-
name>
            <admin-object>
```

```
            <property>
              <prop-name>messageType</prop-name>
              <prop-type>java.lang.String</prop-type>
              <prop-value>Simple</prop-value>
            </property>
          </admin-object>
        </resource-env-ref>
      </message-driven>
      ...
    </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <resource-env-ref> 要素

子要素

- なし

<assembly-descriptor> 要素

```
<element name="assembly-descriptor" minOccurs="0">
 <complexType>
  <sequence>
   <element name="security-role" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="role-name" type="xsd:string"/>
      <element name="deployment-role" type="xsd:string" minOccurs="0"/>
     </sequence>
    </complexType>
   </element>
   <element name="message-destination" minOccurs="0"
maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="message-destination-name" type="xsd:string"/>
      <element name="jndi-name" type="xsd:string"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
```

この要素は、`ejb-jar.xml` 内の同じ要素の上に作成されます。その子ノードを使用して、そのアーカイブ内のモジュールが属している 1 つ以上のセキュリティ ロールに関する情報を提供します。

サンプル

```
<assembly-descriptor>
    <security-role>
        <role-name>administrator</role-name>
        <deployment-role>administrator</deployment-role>
    </security-role>
</assembly-descriptor>
```

関連要素

親要素

- <ejb-jar> 要素

子要素

- <security-role> 要素
- <message-destination> 要素

<authorization-domain> 要素

<xsd: element name="authorization-domain" type="xsd:string" minOccurs="0"/>

アーカイブが属する認証ドメインの名前。

サンプル

<authorization-domain>GroupJ</authorization-domain>

関連要素

親要素

- <ejb-jar> 要素

子要素

- なし

<bean-home-name> 要素

<xsd:element name="bean-home-name" type="xsd:string" minOccurs="0"/>

Bean のホーム インターフェイスを参照するために使用される名前を指定します。

サンプル

<bean-home-name>insurance/remote/clerk**</bean-home-name>**

関連要素

親要素

- <session> 要素
- <entity> 要素

子要素

- なし

<bean-local-home-name> 要素

<xsd: element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>

Bean のローカル ホーム インターフェイスを参照するために使用される名前を指定します。

サンプル

<bean-local-home-name>insurance/remote/clerk**</bean-local-home-name>**

関連要素

親要素

- <session> 要素
- <entity> 要素

子要素

- なし

<cascade-delete-db> 要素

<xsd: element name="cascade-delete-db" minOccurs="0">

`<cascade-delete-db>` は、エンティティ Bean オブジェクトを削除する場合に使用します。オブジェクトに対してカスケード削除を指定すると、コンテナは、そのオブジェクトの従属オブジェクトをすべて自動的に削除します。

サンプル

```
<ejb-relation>
 <ejb-relation-name>Customer-Account</ejb-relation-name>
 <ejb-relationship-role>
  <ejb-relationship-role-name>Account-Has-Customer
  </ejb-relationship-role-name>
  <multiplicity>one</multiplicity>
  <cascade-delete/>
 </ejb-relationship-role>
</ejb-relation>
```

関連要素

親要素

- <ejb-relationship-role> 要素

子要素

- なし

<cmp2-info> 要素

```
<element name="cmp2-info">
 <complexType>
  <sequence>
   <element name="cmp-field" minOccurs="0" maxOccurs="unbounded">
   <complexType>
    <sequence>
     <element name="field-name" type="xsd:string"/>
     <choice>
      <element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
      <complexType>
       <sequence>
        <element name="field-name" type="xsd:string"/>
        <element name="column-name" type="xsd:string"/>
       </sequence>
       </complexType>
       </element>
       <element name="column-name" type="xsd:string"/>
      </choice>
      <element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
   <element name="table-name" type="xsd:string"/>
   <element name="table-ref" type="borl:table-refType" minOccurs="0"
maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
```

CMP 2.0 を使用している場合は、`<cmp2-info>` 要素を使用して、エンティティ Bean を管理するコンテナに情報を提供します。この要素とその子ノードには、CMP フィールドをデータベース列にマップするためのすべての情報が含まれています。

サンプル

```
<cmp2-info>
 <cmp-field>
  <field-name>orderNumber</field-name>
  <column-name>ORDER_NUMBER</column-name>
 </cmp-field>
 <cmp-field>
  <field-name>line</field-name>
  <column-name>LINE</column-name>
 </cmp-field>
</cmp2-info>
```

関連要素

親要素

- <entity> 要素

子要素

- <cmp-field> 要素
- <table-name> 要素
- <table-ref> 要素

<cmp-field> 要素

```
<element name="cmp-field" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="field-name" type="xsd:string"/>
   <choice>
    <element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="field-name" type="xsd:string"/>
       <element name="column-name" type="xsd:string"/>
      </sequence>
     </complexType>
    </element>
    <element name="column-name" type="xsd:string"/>
   </choice>
   <element name="property" type="borl:propertyType" minOccurs=
        "0" maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
```

<cmp-field> 要素を使用して、基本的なフィールド マッピングが行われます。この要素の子ノードでは、フィールド名とマップ先の対応列を指定します。一般には粗粒度のエンティティ Bean で Java クラスを実装して細粒度のデータを表しています。3 番目の子ノード <cmp-field-map> は、細粒度クラスとその規定のデータベース表現とのフィールド マップを定義し、<column-name> 要素のかわりに使用できます。

サンプル

```
<cmp-field>
  <field-name>orderNumber</field-name>
  <column-name>ORDER_NUMBER</column-name>
</cmp-field>
```

関連要素

親要素

- <cmp2-info> 要素

子要素

- <field-name> 要素
- <column-name> 要素
- <cmp-field-map> 要素
- <property> 要素

<cmp-field-map> 要素

```
<element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="field-name" type="xsd:string"/>
   <element name="column-name" type="xsd:string"/>
  </sequence>
 </complexType>
</element>
```

`<cmp-field-map>` 要素は、細粒度 Java クラスとその基底のデータベース表現とのフィールド マップを定義します。このようなクラスでは、`java.io.Serializable` を実装するものとし、そのすべてのデータ メンバーはパブリックであるものとします。

サンプル

```
<cmp-field>
  <field-name>Address</field-name>
  <cmp-field-map>
   <field-name>Address.AddressLine</field-name>
   <column-name>STREET</column-name>
  </cmp-field-map>
  <cmp-field-map>
  ...
 </cmp-field>
```

関連要素

親要素

- <cmp-field> 要素

子要素

- <field-name> 要素
- <column-name> 要素

<cmp-info> 要素

```
<element name="cmp-info">
 <complexType>
  <sequence>
   <element name="description" type="xsd:string" minOccurs="0"/>
   <element name="database-map" minOccurs="0">
    <complexType>
     <sequence>
      <element name="table" type="xsd:string" minOccurs="0"/>
      <element name="column-map" minOccurs="0" maxOccurs="unbounded">
       <complexType>
        <sequence>
         <element name="field-name" type="xsd:string"/>
         <element name="column-name" type="xsd:string" minOccurs="0"/>
         <element name="column-type" type="xsd:string" minOccurs="0"/>
         <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
        </sequence>
       </complexType>
      </element>
     </sequence>
    </complexType>
   </element>
   <element name="finder" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="method-signature" type="xsd:string"/>
      <element name="where-clause" type="xsd:string"/>
      <element name="load-state" type="xsd:string" minOccurs="0"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
```

`<cmp-info>` 要素を使用して、CMP 1.x エンティティ Bean に関する情報を提供します（CMP 2.x を使用している場合は、<cmp2-info> 要素 を使用します）。この要素には 3 つの子ノード、`<description>`、`<database-map>`、`<finder>` があり、Bean の背後にあるデータベースへアクセスし、適切なクエリを使用するために必要なデータを指定します。

サンプル

```
<cmp-info>
    <description/>
    <database-map>
    <table>Courses</table>
    <column-map>
    <field-name>students</field-name>
    <ejb-ref-name>ejb/Student.findByCourse</ejb-ref-name>
    </column-map>
    </database-map>
    <finder>
        <method-signature>findByStudent(Student s)</method-signature>
        <where-clause>SELECT course_dept, course_number FROM
            Enrollment WHERE student  = :s[ejb/Student]</where-clause>
        <load-state>False</load-state>
    </finder>
</cmp-info>
```

関連要素

親要素

- <entity> 要素

子要素

- <description> 要素
- <database-map> 要素
- <finder> 要素

<cmp-resource> 要素

<xsd: element name="cmp-resource" type="xsd:string" minOccurs="0"/>

参照先のリソースが container-managed-persistence を使用する場合は、このフラグを `True` に設定します。この要素は CMP 1.x リソースでのみ有効です。このフラグが設定されている場合、BAS は、標準の `ejb-jar.xml` デプロイメント ディスクリプタ内にある対応するリソース リファレンスを無視します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         <ejb-name>checking</ejb-name>
         <bean-home-name>bank/remote/accounts/checking</bean-home-name>
         <resource-ref>
            <res-ref-name>jdbc/CheckingDataSource</res-ref-name>
            <jndi-name>serial://datasources/Oracle</jndi-name>
            <cmp-resource>True</cmp-resource>
         </resource-ref>
         <cmp-info>
            <database-map>
               <table>Checking_Accounts</table>
            </database-map>
            <finder>
               <method-signature>findAccountsLargerThan
                  (float balance)</method-signature>
               <where-clause>balance &gt; :balance</where-clause>
            </finder>
         </cmp-info>
      </entity>
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <resource-ref> 要素

子要素

- なし

<cmr-field> 要素

```
<element name="cmr-field" minOccurs="0">
  <complexType>
    <sequence>
      <element name="cmr-field-name" type="xsd:string"/>
      <element name="table-ref" type="borl:table-refType"/>
      <element name="property" type="borl:propertyType" minOccurs=
              "0" maxOccurs="unbounded"/>
    </sequence>
  </complexType>
</element>
```

エンティティが別のエンティティへのマップに使用するフィールドと、それに付随する基底のテーブル マッピングを定義します。

サンプル

```
<cmr-field>
   <cmr-field-name>specialInformation</cmr-field-name>
   <table-ref>
    <left-table>
     <table-name>CUSTOMER</table-name>
     <column-list>CUSTOMER_NO</column-list>
    </left-table>
    <right-table>
     <table-name>SPECIAL_INFO</table-name>
     <column-list>CUSTOMER_NO</column-list>
    </right-table>
   </table-ref>
</cmr-field>
```

関連要素

親要素

- <ejb-relationship-role> 要素

子要素

- <cmr-field-name> 要素
- <table-ref> 要素
- <property> 要素

<cmr-field-name> 要素

```
<xsd: element name="cmr-field-name" type="xsd:string"/>
```

エンティティが、関係を保持する別のエンティティへのマップに使用するフィールド。

サンプル

```
<cmr-field-name>specialInformation</cmr-field-name>
```

関連要素

親要素

- <cmr-field> 要素

子要素

- なし

<column-list> 要素

```
<complexType name="column-listType">
  <sequence>
   <element name="column-name" type="xsd:string" maxOccurs="unbounded"/>
  </sequence>
 </complexType>

<xsd:element name="column-list" type="borl:column-listType"/>
```

別のテーブルの列にマップされる、テーブルの列を指定します。各列は、子ノード `<column-name>` で表されます。

サンプル

```
<column-list>
    <column-name>EMP_NO</column-name>
    <column-name>LAST_NAME</column-name>
    <column-name>PROJ_ID</column-name>
</column-list>
```

関連要素

親要素

- <right-table> 要素
- <left-table> 要素
- <cross-table> 要素

子要素

- <column-name> 要素

<column-map> 要素

```
<element name="column-map" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="field-name" type="xsd:string"/>
   <element name="column-name" type="xsd:string" minOccurs="0"/>
   <element name="column-type" type="xsd:string" minOccurs="0"/>
   <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
</element>
```

この要素は、エンティティ Bean が使用するデータベース列に関する情報を CMP 1.x エンジンに提供するために使用されます。エンティティ Bean のフィールド名のほか、そのフィールドに対応する列の情報またはその列を表す EJB リファレンスの名前のどちらかを提供します。

サンプル

```
<column-map>
    <field-name>students</field-name>
    <ejb-ref-name>ejb/Student.findByCourse</ejb-ref-name>
</column-map>
```

関連要素

親要素

- <database-map> 要素

子要素

- <field-name> 要素
- <column-name> 要素
- <column-type> 要素
- <ejb-ref-name> 要素

<column-name> 要素

```
<xsd:element name="column-name" type="xsd:string" maxOccurs="unbounded"/>

<xsd:element name="column-name" type="xsd:string" minOccurs="0"/>

<xsd: element name="column-name" type="xsd:string"/>
```

エンティティ マッピングまたはプロパティ設定のためのデータベース列の名前を指定します。

サンプル

```
<column-name>course_dept</column-name>
```

関連要素

親要素

- <field-name> 要素
- <cmp-field-map> 要素
- <column-list> 要素
- <column-map> 要素
- <column-properties> 要素

子要素

- なし

<column-properties> 要素

```
<element name="column-properties" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="column-name" type="xsd:string"/>
   <element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
```

この要素を使用して、データベース テーブル内の列に固有のプロパティを指定します。この要素の子ノードで、列名や関連するプロパティを指定します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
     <entity>
        <ejb-name>claim</ejb-name>
        <bean-local-home-name>Claim</bean-local-home-name>
        <cmp2-info>
           <cmp-field>
              <field-name>claimId</field-name>
              <column-name>CLAIM_ID</column-name>
           </cmp-field>
           <cmp-field>
              <field-name>policyHolderNumber</field-name>
              <column-name>POLICYHOLDER_NUMBER</column-name>
           </cmp-field>
           <table-name>CLAIMS</table-name>
        </cmp2-info>
     </entity>
     ...
   </enterprise-beans>
   <table-properties>
        <table-name>CLAIMS</table-name>
        <column-properties>
          <column-name>CLAIM_ID</column-name>
          <property>
             <prop-name>createColumnSql</prop-name>
             <prop-type>String</prop-type>
             <prop-value>VARCHAR(10)</prop-value>
          </property>
        </column-properties>
        <column-properties>
          <column-name>POLICYHOLDER_NUMBER</column-name>
          <property>
             <prop-name>createColumnSql</prop-name>
             <prop-type>String</prop-type>
             <prop-value>INT</prop-value>
          </property>
        </column-properties>
        <property>
           <prop-name>datasource</prop-name>
           <prop-type>String</prop-type>
           <prop-value>datasources/insurance/XADataSource</prop-value>
        </property>
   </table-properties>
</ejb-jar>
```

関連要素

親要素

- <table-properties> 要素

子要素

- <column-name> 要素
- <property> 要素

<column-type> 要素

```
<xsd: element name="column-type" type="xsd:string" minOccurs="0"/>
```

エンティティ Bean の CMP クエリの一部としてデータベース列に格納されているデータの型を指定します。

サンプル

```
<column-type>integer</column-type>
```

関連要素

親要素

- <column-type> 要素

子要素

- なし

<connection-factory-name> 要素

```
<xsd: element name="connection-factory-name" type="xsd:string"/>
```

Bean が JMS サービスに接続するために使用する JMS トピックまたはキューの接続ファクトリの JNDI 名。

サンプル

```
<ejb-jar>
    <enterprise-beans>
        <message-driven>
            <ejb-name>MessageReflectorEJB</ejb-name>
            <message-source>
                <jms-provider-ref>
                <message-driven-destination-name>serial://jms/q</message-driven-
destination-name>
                <connection-factory-name>serial://jms/xacf</connection-factory-
name>
                <pool>
                    <max-size>20</max-size>
                    <init-size>1</init-size>
                </pool>
                </jms-provider-ref>
            </message-source>
            <resource-ref>
                <res-ref-name>jms/ConnectionFactory</res-ref-name>
                <jndi-name>serial://jms/xacf</jndi-name>
            </resource-ref>
            <resource-env-ref>
                <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
                <jndi-name>serial://jms/t</jndi-name>
            </resource-env-ref>
        </message-driven>
        ...
    </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <jms-provider-ref> 要素

子要素

- なし

<cross-table> 要素

```
<element name="cross-table" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="table-name" type="xsd:string"/>
   <element name="column-list" type="borl:column-listType"/>
   <element name="column-list" type="borl:column-listType"/>
  </sequence>
 </complexType>
</element>
```

多対多の関係を定義する場合、 CMP エンジンに、 左右テーブルの関係をモデル化する、 クロステーブルを作成させなければなりません。これを行う際に `<cross-table>` 要素を使用します。 クロステーブルには、 子ノードの `<table-name>` 要素で好きな名前をつけることができます。 2 つの子要素 `<column-list>` は、 関係モデルを作成する左右テーブルの列に相当します。

サンプル

```
<table-ref>
    <left-table>
        <table-name>EMPLOYEE</table-name>
        <column-list>
            <column-name>EMP_NO</column-name>
            <column-name>LAST_NAME</column-name>
            <column-name>PROJ_ID</column-name>
        </column-list>
    </left-table>
    <cross-table>
        <table-name>EMPLOYEE_PROJECTS</table-name>
        <column-list>
            <column-name>EMP_NAME</column-name>
            <column-name>PROJ_ID</column-name>
        </column-list>
        <column-list>
            <column-name>PROJ_ID</column-name>
            <column-name>PROJ_NAME</column-name>
        </column-list>
    </cross-table>
    <right-table>
        <table-name>PROJECT</table-name>
        <column-list>
            <column-name>PROJ_ID</column-name>
            <column-name>PROJ_NAME</column-name>
            <column-name>EMP_NO</column-name>
        </column-list>
    </right-table>
</table-ref>
```

関連要素

親要素

- <table-ref> 要素

子要素

- <table-name> 要素
- <column-list> 要素

<database-map> 要素

```
<element name="database-map" minOccurs="0">
 <complexType>
  <sequence>
   <element name="table" type="xsd:string" minOccurs="0"/>
   <element name="column-map" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="field-name" type="xsd:string"/>
      <element name="column-name" type="xsd:string" minOccurs="0"/>
      <element name="column-type" type="xsd:string" minOccurs="0"/>
      <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
```

<database-map> 要素は、 エンティティ Bean のフィールドにデータを格納したり、 検索メソッドを実行するために CMP 1.x エンジンが必要とするデータソース情報を提供するために使用されます。その子ノードで、 使用するデータベース テーブルを指定します。 また、 エンティティ Bean が自分のフィールドにデータを格納するために使用するフィールドと列も指定します。

サンプル

```
<database-map>
      <table>Courses</table>
      <column-map>
        <field-name>students</field-name>
        <ejb-ref-name>ejb/Student.findByCourse</ejb-ref-name>
      </column-map>
</database-map>
```

関連要素

親要素

- <cmp-info> 要素

子要素

- <table> 要素
- <column-map> 要素

<datasource-definitions> 要素

```
<element name="datasource-definitions" minOccurs="0">
 <complexType>
  <sequence>
   <element name="datasource" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="jndi-name" type="xsd:string"/>
      <element name="url" type="xsd:string"/>
      <element name="username" type="xsd:string" minOccurs="0"/>
      <element name="password" type="xsd:string" minOccurs="0"/>
      <element name="isolation-level" type="xsd:string" minOccurs="0"/>
      <element name="driver-class-name" type="xsd:string" minOccurs="0"/>
      <element name="jdbc-property" minOccurs="0" maxOccurs="unbounded">
       <complexType>
        <sequence>
         <element name="prop-name" type="xsd:string"/>
         <element name="prop-value" type="xsd:string"/>
        </sequence>
       </complexType>
      </element>
      <element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
```

古い形式の JDBC 1.x データソースを使用している場合に、この要素は、アーカイブ内の Bean が使用するデータソースに関する情報を提供するために使用されます。各データソースは、個別の `<datasource>` 要素内で定義されます。これは、`<datasource-definitions>` の子ノードです。通常は、新しい形式の `jndi-definitions.xml` ファイルを使ってデータソースを定義します。この要素は JDBC 1.x のユーザー専用です。

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
        <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver</driver-
class-name>
     </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <ejb-jar> 要素

子要素

- <datasource> 要素

<datasource> 要素

```
<element name="datasource" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="jndi-name" type="xsd:string"/>
   <element name="url" type="xsd:string"/>
   <element name="username" type="xsd:string" minOccurs="0"/>
   <element name="password" type="xsd:string" minOccurs="0"/>
   <element name="isolation-level" type="xsd:string" minOccurs="0"/>
   <element name="driver-class-name" type="xsd:string" minOccurs="0"/>
   <element name="jdbc-property" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="prop-name" type="xsd:string"/>
      <element name="prop-value" type="xsd:string"/>
     </sequence>
    </complexType>
   </element>
   <element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
```

この要素は、JDBC 1.x データソースを記述するために使用されます。通常は、新しい形式の `jndi-definitions.xml` ファイルを使ってデータソースを定義します。この要素とその子ノードは JDBC 1.x にだけ適用されます。

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
        <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver
                </driver-class-name>
     </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <datasource-definitions> 要素

子要素

- <jndi-name> 要素
- <url> 要素
- <username> 要素
- <password> 要素
- <isolation-level> 要素
- <driver-class-name> 要素
- <jdbc-property> 要素
- <property> 要素

<deployment-role> 要素

<xsd: element name="deployment-role" type="xsd:string" minOccurs="0"/>

アーカイブ内のモジュールによって使用されるデプロイメント ロールのロール名。

サンプル

```
<deployment-role>administrator</deployment-role>
```

関連要素

親要素

- <security-role> 要素

子要素

- なし

<description> 要素

<xsd: element name="description" type="xsd:string" minOccurs="0"/>

このオプションの要素を使用して、親ノードの説明を記述します。

サンプル

```
<description>sorting bean</description>
```

関連要素

親要素

- <cmp-info> 要素

子要素

- なし

<driver-class-name> 要素

```
<xsd: element name="driver-class-name" type="xsd:string" minOccurs="0"/>
```

JDBC 1.x 専用。定義されるデータソースにアクセスするために使用されるドライバのクラス名。

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
        <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver
              </driver-class-name>
     </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <datasource> 要素

子要素

- なし

<ejb-jar> 要素

```
<complexType name="ejb-jarType">
 <sequence>
  <element name="enterprise-beans">
   <complexType>
    <choice maxOccurs="unbounded">
     <element name="session">
      <complexType>
       <sequence>
        <element name="ejb-name" type="xsd:string"/>
        <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
        <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
        <element name="timeout" type="xsd:string" minOccurs="0"/>
        <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
        <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
       </sequence>
      </complexType>
     </element>
     <element name="entity">
      <complexType>
       <sequence>
        <element name="ejb-name" type="xsd:string"/>
        <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
        <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
        <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
        <choice minOccurs="0">
         <element name="cmp-info">
          <complexType>
           <sequence>
            <element name="description" type="xsd:string" minOccurs="0"/>
            <element name="database-map" minOccurs="0">
             <complexType>
              <sequence>
               <element name="table" type="xsd:string" minOccurs="0"/>
               <element name="column-map" minOccurs="0" maxOccurs="unbounded">
                <complexType>
                 <sequence>
                  <element name="field-name" type="xsd:string"/>
                  <element name="column-name" type="xsd:string" minOccurs="0"/>
                  <element name="column-type" type="xsd:string" minOccurs="0"/>
                  <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
                 </sequence>
                </complexType>
               </element>
              </sequence>
             </complexType>
            </element>
            <element name="finder" minOccurs="0" maxOccurs="unbounded">
             <complexType>
              <sequence>
               <element name="method-signature" type="xsd:string"/>
```

```
      <element name="where-clause" type="xsd:string"/>
      <element name="load-state" type="xsd:string" minOccurs="0"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
<element name="cmp2-info">
 <complexType>
  <sequence>
   <element name="cmp-field" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="field-name" type="xsd:string"/>
       <choice>
        <element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
         <complexType>
          <sequence>
           <element name="field-name" type="xsd:string"/>
           <element name="column-name" type="xsd:string"/>
          </sequence>
         </complexType>
        </element>
        <element name="column-name" type="xsd:string"/>
       </choice>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
    <element name="table-name" type="xsd:string"/>
    <element name="table-ref" type="borl:table-refType" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
</choice>
<element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
<element name="query" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="query-method">
    <complexType>
     <sequence>
      <element name="method-name" type="xsd:string"/>
      <element name="method-params">
       <complexType>
        <sequence>
         <element name="method-param" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
       </complexType>
      </element>
     </sequence>
    </complexType>
   </element>
   <element name="user-sql" type="xsd:string" minOccurs="0"/>
   <element name="load-state" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
</element>
</sequence>
</complexType>
</element>
```

```
    <element name="message-driven">
     <complexType>
      <sequence>
       <element name="ejb-name" type="xsd:string"/>
       <element name="message-source">
        <complexType>
         <choice>
          <element name="resource-adapter-ref" type="borl:resource-adapter-refType"/>
          <element name="jms-provider-ref" type="borl:jms-provider-refType"/>
         </choice>
        </complexType>
       </element>
       <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
       <element name="resource-env-ref" type="borl:resource-env-refMdbType" minOccurs="0"
maxOccurs="unbounded"/>
       <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
   </choice>
  </complexType>
 </element>
 <element name="datasource-definitions" minOccurs="0">
  <complexType>
   <sequence>
    <element name="datasource" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="jndi-name" type="xsd:string"/>
       <element name="url" type="xsd:string"/>
       <element name="username" type="xsd:string" minOccurs="0"/>
       <element name="password" type="xsd:string" minOccurs="0"/>
       <element name="isolation-level" type="xsd:string" minOccurs="0"/>
       <element name="driver-class-name" type="xsd:string" minOccurs="0"/>
       <element name="jdbc-property" minOccurs="0" maxOccurs="unbounded">
        <complexType>
         <sequence>
          <element name="prop-name" type="xsd:string"/>
          <element name="prop-value" type="xsd:string"/>
         </sequence>
        </complexType>
       </element>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
   </sequence>
  </complexType>
 </element>
 <element name="table-properties" minOccurs="0" maxOccurs="unbounded">
  <complexType>
   <sequence>
    <element name="table-name" type="xsd:string"/>
    <element name="column-properties" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="column-name" type="xsd:string"/>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
```

```
      </sequence>
     </complexType>
    </element>
    <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
 <element name="relationships" minOccurs="0">
  <complexType>
   <sequence>
    <element name="ejb-relation" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="ejb-relationship-role" type="borl:ejb-relationship-roleType"/>
       <element name="ejb-relationship-role" type="borl:ejb-relationship-roleType"/>
      </sequence>
     </complexType>
    </element>
   </sequence>
  </complexType>
 </element>
 <element name="authorization-domain" type="xsd:string" minOccurs="0"/>
 <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
 <element name="assembly-descriptor" minOccurs="0">
  <complexType>
   <sequence>
    <element name="security-role" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="role-name" type="xsd:string"/>
       <element name="deployment-role" type="xsd:string" minOccurs="0"/>
      </sequence>
     </complexType>
    </element>
    <element name="message-destination" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="message-destination-name" type="xsd:string"/>
       <element name="jndi-name" type="xsd:string"/>
      </sequence>
     </complexType>
    </element>
   </sequence>
  </complexType>
 </element>
</sequence>
</complexType>
<xsd:element name="ejb-jar" type="borl:ejb-jarType"/>
```

`<ejb-jar>` 要素は、`ejb-borland.xml` のルート ノードです。その子ノードは、JAR によってデプロイされるエンタープライズ Bean を定義し、Bean 間の関係に関する情報を提供します。また、Bean が使用するデータソース（データベース、JMS プロバイダなど）に関する情報のほか、アーカイブ プロパティやセキュリティ関連情報を指定することもできます。

サンプル

```
<ejb-jar>
    <enterprise-beans>
        <session>
            <ejb-name>AsyncSenderEJB</ejb-name>
            <bean-local-home-name>ejb/local/petstore/asyncsender/AsyncSender
```

```
            </bean-local-home-name>
          <timeout>0</timeout>
          <resource-ref>
            <res-ref-name>jms/queue/QueueConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xaqcf</jndi-name>
          </resource-ref>
          <resource-env-ref>
            <resource-env-ref-name>jms/queue/AsyncSenderQueue
              </resource-env-ref-name>
            <jndi-name>serial://jms/queue/opc/OrderQueue</jndi-name>
          </resource-env-ref>
        </session>
    </enterprise-beans>
    <assembly-descriptor/>
</ejb-jar>
```

関連要素

親要素

- なし

子要素

- <enterprise-beans> 要素
- <datasource-definitions> 要素
- <table-properties> 要素
- <relationships> 要素
- <authorization-domain> 要素
- <property> 要素
- <assembly-descriptor> 要素

<ejb-local-ref> 要素

```
<xsd: element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0"
maxOccurs="unbounded"/>

<complexType name="ejb-local-refType">
 <sequence>
  <element name="ejb-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string" minOccurs="0"/>
 </sequence>
</complexType>
```

ejb-local-ref 要素は、EJB コンテナによってローカルで解決される EJB リファレンスを表します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         <ejb-name>clerk</ejb-name>
         <bean-home-name>insurance/remote/clerk</bean-home-name>
         <timeout>5</timeout>
         <ejb-local-ref>
            <ejb-ref-name>ejb/insurance/claim</ejb-ref-name>
         </ejb-local-ref>
         <resource-ref>
                  <res-ref-name>jms/insurance/ConnectionFactory</res-ref-name>
                  <jndi-name>jms/xacf</jndi-name>
```

```
            </resource-ref>
        </session>
        <entity>
            <ejb-name>claim</ejb-name>
            <bean-local-home-name>Claim</bean-local-home-name>
            ...
        </entity>
        ...
    </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <session> 要素

子要素

- なし

<ejb-name> 要素

```
<xsd: element name="ejb-name" type="xsd:string"/>
```

<ejb-name> 要素を使用して、定義するエンタープライズ JavaBeans の名前を指定します。この要素は、ejb-jar.xml 内の同じ要素と同様に、リモートからの Bean のルックアップに使用する名前を指定します。

サンプル

```
<ejb-name>clerk</ejb-name>
```

関連要素

親要素

- <session> 要素
- <entity> 要素
- <message-driven> 要素
- <relationship-role-source> 要素

子要素

- なし

<ejb-ref> 要素

```
<xsd: element name="ejb-ref" type="borl:ejb-refType" minOccurs="0"
maxOccurs="unbounded"/>

<complexType name="ejb-refType">
  <sequence>
   <element name="ejb-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
```

この要素は、Bean によって使用される EJB リファレンスを定義するために使用されます。各 EJB リファレンスには、クライアント アプリケーションによって使用される ejb-ref-name と、それに関連付けられた jndi-name （該当する場合）が含まれています。

サンプル

```
<ejb-ref>
    <ejb-ref-name>ejb/Sort</ejb-ref-name>
    <jndi-name>sort</jndi-name>
</ejb-ref>
```

関連要素

親要素

- <session> 要素
- <entity> 要素
- <message-driven> 要素

子要素

- <ejb-ref-name> 要素
- <jndi-name> 要素

<ejb-ref-name> 要素

```
<xsd: element name="ejb-ref-name" type="xsd:string"/>

<xsd: element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
```

この要素は、Bean によってリソース リファレンスとして使用される EJB の名前を提供します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         <ejb-name>clerk</ejb-name>
         <bean-home-name>insurance/remote/clerk</bean-home-name>
         <timeout>5</timeout>
         <ejb-local-ref>
            <ejb-ref-name>ejb/insurance/claim</ejb-ref-name>
         </ejb-local-ref>
         <resource-ref>
                  <res-ref-name>jms/insurance/ConnectionFactory</res-ref-name>
                  <jndi-name>jms/xacf</jndi-name>
         </resource-ref>
      </session>
      <entity>
          <ejb-name>claim</ejb-name>
          <bean-local-home-name>Claim</bean-local-home-name>
          ...
      </entity>
      ...
    </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <column-map> 要素

子要素

- なし

<ejb-relation> 要素

```
<xsd:element name="ejb-relation" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ejb-relationship-role" type="borl:
            ejb-relationship-roleType"/>
      <xsd:element name="ejb-relationship-role" type="borl:
            ejb-relationship-roleType"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

この要素は、2 つのエンティティの関係を表します。各エンティティ Bean からもう一方への Bean への関係は、それぞれ個別に子ノード `<ejb-relationship-role>` を使用して定義されます。これは一方向の関係の場合でも該当します。

サンプル

これらの関係は、以下の例のように、さまざまな形式になります。

- 単一方向の 1 対 1 の関係
- 双方向の 1 対多の関係

単一方向の 1 対 1 の関係

```
<ejb-jar>
  ...
  <relationships>
     <ejb-relation>
       <ejb-relationship-role>
          <relationship-role-source>
             <ejb-name>Customer</ejb-name>
          </relationship-role-source>
          <cmr-field>
             <cmr-field-name>specialInformation</cmr-field-name>
             <table-ref>
                <left-table>
                   <table-name>CUSTOMER</table-name>
                   <column-list>
                      <column-name>CUSTOMER_NO</column-name>
                   </column-list>
                </left-table>
                <right-table>
                   <table-name>SPECIAL_INFO</table-name>
                   <column-list>
                      <column-name>CUSTOMER_NO</column-name>
                   </column-list>
                </right-table>
             </table-ref>
          </cmr-field>
       </ejb-relationship-role>
       <ejb-relationship-role>
          <relationship-role-source>
             <ejb-name>SpecialInfo</ejb-name>
          </relationship-role-source>
       </ejb-relationship-role>
     </ejb-relation>
  </relationships>
  ...
</ejb-jar>
```

この関係は単一方向なので、`SpecialInfo` Bean に対してテーブル情報を指定する必要はありません。

双方向の 1 対多の関係

```
<ejb-jar>
  ...
  <relationships>
     <ejb-relation>
       <ejb-relationship-role>
          <relationship-role-source>
             <ejb-name>Customer</ejb-name>
          </relationship-role-source>
          <cmr-field>
             <cmr-field-name>orders</cmr-field-name>
             <table-ref>
                <left-table>
                   <table-name>CUSTOMER</table-name>
                   <column-list>
                      <column-name>CUSTOMER_NO</column-name>
                   </column-list>
                </left-table>
                <right-table>
                   <table-name>ORDER</table-name>
                   <column-list>
                      <column-name>CUSTOMER_NO</column-name>
                   </column-list>
                </right-table>
             </table-ref>
          </cmr-field>
       </ejb-relationship-role>
       <ejb-relationship-role>
          <relationship-role-source>
             <ejb-name>Order</ejb-name>
          </relationship-role-source>
          <cmr-field>
             <cmr-field-name>customers</cmr-field-name>
             <table-ref>
                <left-table>
                   <table-name>ORDER</table-name>
                   <column-list>
                      <column-name>CUSTOMER_NO</column-name>
                   </column-list>
                </left-table>
                <right-table>
                   <table-name>CUSTOMER</table-name>
                   <column-list>
                      <column-name>CUSTOMER_NO</column-name>
                   </column-list>
                </right-table>
             </table-ref>
          </cmr-field>
        </ejb-relationship-role>
     </ejb-relation>
   </relationships>
   ...
</ejb-jar>
```

このテーブルは両方向にリンクされるため、各方向にテーブル データが提供されます。

関連要素

親要素

- <relationships> 要素

子要素

- <ejb-relationship-role> 要素

<ejb-relationship-role> 要素

```
<xsd:complexType name="ejb-relationship-roleType">
 <xsd:sequence>
  <xsd:element name="relationship-role-source">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="ejb-name" type="xsd:string"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="cmr-field" minOccurs="0">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="cmr-field-name" type="xsd:string"/>
     <xsd:element name="table-ref" type="borl:table-refType"/>
     <xsd:element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="cascade-delete-db" minOccurs="0">
</xsd:sequence>
</xsd:complexType>

<xsd: element name="ejb-relationship-role" type="borl:ejb-relationship-roleType"/>
```

1 つのエンティティを定義し、別のエンティティとの関係も定義します。エンティティ自体は子ノード `<relationship-role-source>` を使って提供され、関係のもう一方のエンティティと共有するフィールドは、子ノード `<cmr-field>` で定義されます。

サンプル

```
<ejb-relationship-role>
  <relationship-role-source>
   <ejb-name>Customer</ejb-name>
  </relationship-role-source>
  <cmr-field>
   <cmr-field-name>specialInformation</cmr-field-name>
   <table-ref>
    <left-table>
     <table-name>CUSTOMER</table-name>
     <column-list>CUSTOMER_NO</column-list>
    </left-table>
    <right-table>
     <table-name>SPECIAL_INFO</table-name>
     <column-list>CUSTOMER_NO</column-list>
    </right-table>
   </table-ref>
  </cmr-field>
</ejb-relationship-role>
```

関連要素

親要素

- <ejb-relation> 要素

子要素

- <relationship-role-source> 要素
- <cmr-field> 要素
- <cascade-delete-db> 要素

<enterprise-beans> 要素

```
<element name="enterprise-beans">
 <complexType>
  <choice maxOccurs="unbounded">
   <element name="session">
    <complexType>
     <sequence>
      <element name="ejb-name" type="xsd:string"/>
      <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
      <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
      <element name="timeout" type="xsd:string" minOccurs="0"/>
      <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
      <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
   <element name="entity">
    <complexType>
     <sequence>
      <element name="ejb-name" type="xsd:string"/>
      <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
      <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
```

```
    <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <choice minOccurs="0">
     <element name="cmp-info">
      <complexType>
       <sequence>
        <element name="description" type="xsd:string" minOccurs="0"/>
        <element name="database-map" minOccurs="0">
         <complexType>
          <sequence>
           <element name="table" type="xsd:string" minOccurs="0"/>
           <element name="column-map" minOccurs="0" maxOccurs="unbounded">
            <complexType>
             <sequence>
              <element name="field-name" type="xsd:string"/>
              <element name="column-name" type="xsd:string" minOccurs="0"/>
              <element name="column-type" type="xsd:string" minOccurs="0"/>
              <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
             </sequence>
            </complexType>
           </element>
          </sequence>
         </complexType>
        </element>
        <element name="finder" minOccurs="0" maxOccurs="unbounded">
         <complexType>
          <sequence>
           <element name="method-signature" type="xsd:string"/>
           <element name="where-clause" type="xsd:string"/>
           <element name="load-state" type="xsd:string" minOccurs="0"/>
          </sequence>
         </complexType>
        </element>
       </sequence>
      </complexType>
     </element>
     <element name="cmp2-info">
      <complexType>
       <sequence>
        <element name="cmp-field" minOccurs="0" maxOccurs="unbounded">
         <complexType>
          <sequence>
           <element name="field-name" type="xsd:string"/>
            <choice>
             <element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
              <complexType>
               <sequence>
                <element name="field-name" type="xsd:string"/>
                <element name="column-name" type="xsd:string"/>
               </sequence>
              </complexType>
             </element>
             <element name="column-name" type="xsd:string"/>
            </choice>
           <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
          </sequence>
         </complexType>
        </element>
```

```
        <element name="table-name" type="xsd:string"/>
        <element name="table-ref" type="borl:table-refType" minOccurs="0" maxOccurs="unbounded"/>
       </sequence>
      </complexType>
     </element>
    </choice>
    <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="query" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element name="query-method">
        <complexType>
         <sequence>
          <element name="method-name" type="xsd:string"/>
          <element name="method-params">
           <complexType>
            <sequence>
             <element name="method-param" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
           </complexType>
          </element>
         </sequence>
        </complexType>
       </element>
       <element name="user-sql" type="xsd:string" minOccurs="0"/>
       <element name="load-state" type="xsd:string" minOccurs="0"/>
      </sequence>
     </complexType>
    </element>
   </sequence>
  </complexType>
 </element>
 <element name="message-driven">
  <complexType>
   <sequence>
    <element name="ejb-name" type="xsd:string"/>
    <element name="message-source">
     <complexType>
      <choice>
       <element name="resource-adapter-ref" type="borl:resource-adapter-refType"/>
       <element name="jms-provider-ref" type="borl:jms-provider-refType"/>
      </choice>
     </complexType>
    </element>
    <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="resource-env-ref" type="borl:resource-env-refMdbType" minOccurs="0"
maxOccurs="unbounded"/>
    <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
 </choice>
</complexType>
</element>
```

`<enterprise-beans>` 要素を使用して、アーカイブ内の Java Bean を定義します。3 種類のエンタープライズ Java Bean（セッション Bean、エンティティ Bean、メッセージ駆動型 Bean）のそれぞれに対応する子ノードがあり、ここでこれらの Bean に関する情報を提

供します。各 Bean のタイプに基づいて、アーカイブ内の Bean ごとにエントリを作成します。

サンプル

```
<enterprise-beans>
    <session>
      <ejb-name>UniqueIdGeneratorEJB</ejb-name>
      ...
    </session>
    <entity>
      <ejb-name>CounterEJB</ejb-name>
      ...
    </entity>
</enterprise-beans>
```

関連要素

親要素

- <ejb-jar> 要素

子要素

- <session> 要素
- <entity> 要素
- <message-driven> 要素

<entity> 要素

```
<element name="entity">
 <complexType>
  <sequence>
   <element name="ejb-name" type="xsd:string"/>
   <element name="bean-home-name" type="xsd:string" minOccurs="0"/>
   <element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
   <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
   <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <choice minOccurs="0">
     <element name="cmp-info">
      <complexType>
       <sequence>
        <element name="description" type="xsd:string" minOccurs="0"/>
        <element name="database-map" minOccurs="0">
         <complexType>
          <sequence>
           <element name="table" type="xsd:string" minOccurs="0"/>
           <element name="column-map" minOccurs="0" maxOccurs="unbounded">
            <complexType>
             <sequence>
              <element name="field-name" type="xsd:string"/>
              <element name="column-name" type="xsd:string" minOccurs="0"/>
              <element name="column-type" type="xsd:string" minOccurs="0"/>
              <element name="ejb-ref-name" type="xsd:string" minOccurs="0"/>
             </sequence>
            </complexType>
```

```
      </element>
     </sequence>
    </complexType>
   </element>
   <element name="finder" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="method-signature" type="xsd:string"/>
      <element name="where-clause" type="xsd:string"/>
      <element name="load-state" type="xsd:string" minOccurs="0"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
<element name="cmp2-info">
 <complexType>
  <sequence>
   <element name="cmp-field" minOccurs="0" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="field-name" type="xsd:string"/>
       <choice>
        <element name="cmp-field-map" minOccurs="0" maxOccurs="unbounded">
         <complexType>
          <sequence>
           <element name="field-name" type="xsd:string"/>
           <element name="column-name" type="xsd:string"/>
          </sequence>
         </complexType>
        </element>
        <element name="column-name" type="xsd:string"/>
       </choice>
      <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
   <element name="table-name" type="xsd:string"/>
   <element name="table-ref" type="borl:table-refType" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
 </complexType>
</element>
</choice>
<element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
<element name="query" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element name="query-method">
    <complexType>
     <sequence>
      <element name="method-name" type="xsd:string"/>
      <element name="method-params">
       <complexType>
        <sequence>
         <element name="method-param" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
       </complexType>
      </element>
     </sequence>
    </complexType>
   </element>
```

```
    <element name="user-sql" type="xsd:string" minOccurs="0"/>
    <element name="load-state" type="xsd:string" minOccurs="0"/>
   </sequence>
  </complexType>
 </element>
</sequence>
</complexType>
</element>
```

<entity> 要素は、アーカイブに含まれるエンティティ Bean に関する情報を提供します。この要素の子ノードを使用して、Bean のさまざまなインターフェイスとリファレンスを指定できます。コンテナ管理の永続性情報のほか、各エンティティ Bean に固有の汎用プロパティも提供できます。

サンプル

```
<entity>
    <ejb-name>claim</ejb-name>
    <bean-local-home-name>Claim</bean-local-home-name>
    <cmp2-info>
      <cmp-field>
        <field-name>claimId</field-name>
        <column-name>CLAIM_ID</column-name>
      </cmp-field>
      <cmp-field>
        <field-name>policyHolderNumber</field-name>
        <column-name>POLICYHOLDER_NUMBER</column-name>
      ...
      <table-name>CLAIMS</table-name>
    </cmp2-info>
</entity>
```

関連要素

親要素

- <enterprise-beans> 要素

子要素

- <ejb-name> 要素
- <bean-home-name> 要素
- <bean-local-home-name> 要素
- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination-ref> 要素
- <cmp-info> 要素
- <cmp2-info> 要素
- <property> 要素
- EJB モジュール：ejb-borland.xml

<field-name> 要素

```
<xsd: element name="field-name" type="xsd:string"/>
```

基底のデータソース内の列にマップされるエンティティ Bean フィールドの名前。

サンプル

```
<field-name>students</field-name>
```

関連要素

親要素

- <cmp-field> 要素
- <cmp-field-map> 要素
- <column-map> 要素

子要素

- なし

<finder> 要素

```
<element name="finder" minOccurs="0" maxOccurs="unbounded">
  <complexType>
    <sequence>
      <element name="method-signature" type="xsd:string"/>
      <element name="where-clause" type="xsd:string"/>
      <element name="load-state" type="xsd:string" minOccurs="0"/>
  </sequence>
  </complexType>
</element>
```

この要素を使用して、エンティティ Bean が使用する検索メソッドを定義します。検索メソッドを生成する場合、実際には、where 節を持つ SQL select 文を生成することになります。select 文には、どのレコードまたはデータを検索して返すかを指定する節があります。コンテナ管理による永続性の下では、`<finder>` の子ノードを使用して、where 節の条件を指定する必要があります。

サンプル

```
<finder>
   <method-signature>findByStudent(Student s)</method-signature>
   <where-clause>SELECT course_dept, course_number FROM
            Enrollment WHERE student  = :s[ejb/Student]</where-clause>
   <load-state>False</load-state>
</finder>
```

関連要素

親要素

- <cmp-info> 要素

子要素

- <method-signature> 要素
- <where-clause> 要素
- <load-state> 要素

<init-size> 要素

```
<xsd: element name="init-size" type="xsd:string" minOccurs="0"/>
```

MDB プールが最初に作成されたときに、BAS がプールを満たすために使用する接続の数です。これは `ejb.mdb.init-size` プロパティと同じです。

サンプル

```
<ejb-jar>
   <enterprise-beans>
   <message-driven>
      <ejb-name>MessageReflectorEJB</ejb-name>
      <message-source>
      <jms-provider-ref>
            <message-driven-destination-name>serial://jms/q</message-driven-
destination-name>
         <connection-factory-name>serial://jms/xacf</connection-factory-name>
         <pool>
            <max-size>20</max-size>
            <init-size>1</init-size>
```

```
            <wait-timeout>20</wait-timeout>
          </pool>
        </jms-provider-ref>
        </message-source>
        <resource-ref>
          <res-ref-name>jms/ConnectionFactory</res-ref-name>
          <jndi-name>serial://jms/xacf</jndi-name>
        </resource-ref>
        <resource-env-ref>
          <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
          <jndi-name>serial://jms/t</jndi-name>
        </resource-env-ref>
      </message-driven>
      ...
      </enterprise-beans>
  </ejb-jar>
```

関連要素

親要素

- <pool> 要素

子要素

- なし

<instance-name> 要素

```
<element name="instance-name" type="xsd:string"/>
```

resource-adapter-ref 要素の instance-name は、特定のリソース アダプタを親 MDB によって消費されるメッセージのソースとして識別します。この値は、デプロイされたリソース アダプタの Borland リソース アダプタ ディスクリプタにおける、resourceadapter 要素の instance-name の値と対応します。メッセージ エンドポイントは、エンドポイントの有効化時に、リソース アダプタとバインドされます。

サンプル

```
<ejb-jar>
  <enterprise-beans>
    <message-driven>
      <ejb-name>MessageReflectorEJB</ejb-name>
      <message-source>
        <resource-adapter-ref>
          <instance-name>mailAdapter</instance-name>
        </resource-adapter-ref>
      </message-source>
    </message-driven>
    ...
  </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <resource-adapter-ref> 要素

子要素

- なし

<isolation-level> 要素

<xsd: element name="isolation-level" type="xsd:string" minOccurs="0"/>

JDBC 1.x 専用。定義されるデータソースの分離レベル。次の値の 1 つを指定できます。

- TRANSACTION_NONE
- TRANSACTION_READ_COMMITTED
- TRANSACTION_READ_UNCOMMITTED
- TRANSACTION_REPEATABLE_READ
- TRANSACTION_SERIALIZEABLE

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
       <isolation-level>TRANSACTION_READ_UNCOMMITTED</isolation-level>
        <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver
                </driver-class-name>
     </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <datasource> 要素

子要素

- なし

<jdbc-property> 要素

```
<xsd:element name="jdbc-property" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="prop-name" type="xsd:string"/>
   <xsd:element name="prop-value" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

JDBC 1.x 専用。定義されるデータソースとともに使用される JDBC プロパティを指定します。子ノード `<prop-name>` と `<prop-value>` に対して、それぞれプロパティ名とその値を指定します。

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
       <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver
                  </driver-class-name>
       <jdbc-property>
          <prop-name>connection-timeout</prop-name>
          <prop-value>200</prop-value>
       </jdbc-property>
      </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <datasource> 要素

子要素

- <prop-name> 要素
- <prop-value> 要素

<jms-provider-ref> 要素

```
<xsd:complexType name="jms-provider-refType">
 <xsd:sequence>
  <xsd:element name="message-driven-destination-name" type="xsd:string"/>
  <xsd:element name="connection-factory-name" type="xsd:string"/>
  <xsd:element name="pool" minOccurs="0">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="max-size" type="xsd:string" minOccurs="0"/>
     <xsd:element name="init-size" type="xsd:string" minOccurs="0"/>
     <xsd:element name="wait-timeout" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>
```

<element name="jms-provider-ref" type="borl:jms-provider-refType"/>

この要素は、EJB 2.0 仕様に準じた MDB のアクティベーションを表します。情報は、javax.jms.MessageListener を実装した MDB に提供されます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
              <message-driven-destination-name>serial://jms/q</
                              message-driven-destination-name>
              <connection-factory-name>serial://jms/xacf</
                              connection-factory-name>
              <pool>
                 <max-size>20</max-size>
                 <init-size>1</init-size>
              </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <message-source> 要素

子要素

- <message-driven-destination-name> 要素
- <connection-factory-name> 要素
- <pool> 要素

<jndi-name> 要素

```
<xsd: element name="jndi-name" type="xsd:string"/>
```

この要素は、Bean によって参照されるリソースの、JNDI サービス ルックアップ名を提供します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         <ejb-name>clerk</ejb-name>
         <bean-home-name>insurance/remote/clerk</bean-home-name>
         <timeout>0</timeout>
         <ejb-local-ref>
            <ejb-ref-name>ejb/insurance/claim</ejb-ref-name>
         </ejb-local-ref>
         <resource-ref>
            <res-ref-name>jms/insurance/ConnectionFactory</res-ref-name>
            <jndi-name>jms/xacf</jndi-name>
         </resource-ref>
      </session>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination-ref> 要素
- <datasource> 要素
- <message-destination> 要素

子要素

- なし

<left-table> 要素

```
<element name="left-table">
 <complexType>
  <sequence>
   <element name="table-name" type="xsd:string"/>
   <element name="column-list" type="borl:column-listType"/>
  </sequence>
 </complexType>
</element>
```

<cmp2-info> を指定する場合、この要素は、1 つの列を共有する 2 つのテーブルの一方を定義します。一方がもう一方の外部キーになります。

この要素を使って <relationships> を記述する場合、<left-table> が関係の元、<right-table> が関係の先となります。つまり、関係の方向は左から右です。双方向の関係を定義する場合は、各方向ごとに 1 つ、つまり 1 つの関係に対して 2 つの <table-ref> 要素を作成する必要があります。多対多の関係では、<left-table> は、共通部分を定義する <cross-table> を作成するために使用される 2 つのテーブルの一方を作成します。

サンプル

```
<left-table>
    <table-name>CUSTOMER</table-name>
    <column-list>CUSTOMER_NO</column-list>
</left-table>
```

関連要素

親要素

- <table-ref> 要素

子要素

- <table-name> 要素
- <column-list> 要素

<load-state> 要素

```
<xsd: element name="load-state" type="xsd:string"/>
```

ejbCreate() メソッドの呼び出し時に、コンテナにエンティティ Bean の現在の状態をロードする場合は、このフラグを true に設定します。

サンプル

```
<ejb-jar>
  <enterprise-beans>
    <entity>
      ...
      <query>
        <query-method>
          <method-name>findByStudent</method-name>
          <method-params>
            <method-param>java.lang.String</method-param>
          </method-params>
        </query-method>
        <user-sql>SELECT course_dept, course_number FROM Enrollment
                  i WHERE i.student=?1</user-sql>
        <load-state>True</load-state>
      </query>
    </entity>
    ...
  </enterprise-beans>
  ...
</ejb-jar>
```

関連要素

親要素

- <finder> 要素

- EJB モジュール：ejb-borland.xml

子要素

- なし

<max-size> 要素

<xsd: element name="max-size" type="xsd:string" minOccurs="0"/>

これは、MDB プール内の最大の接続数で、`ejb.mdb.max-size` プロパティと同じです。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
               <message-driven-destination-name>serial://jms/q</message-driven-
destination-name>
               <connection-factory-name>serial://jms/xacf</connection-factory-
name>
               <pool>
                  <max-size>20</max-size>
                  <init-size>1</init-size>
                  <wait-timeout>20</wait-timeout>
               </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <pool> 要素

子要素

- なし

<message-destination> 要素

<element name="message-destination" type="borl:message-destinationType"
minOccurs="0" maxOccurs="unbounded"/>

<complexType name="message-destinationType">

```
    <sequence>
      <element name="message-destination-name" type="xsd:string"/>
      <element name="jndi-name" type="xsd:string"/>
    </sequence>
  </complexType>
```

この要素は、アプリケーション コンポーネントの標準ディスクリプタ内にある 1 つ以上の `message-destination-ref` または `message-driven` 要素の `message-destination-link` に対応する、JMS キューやトピックなどのメッセージの送信先を定義するために使用されます。各メッセージの送信先には、`message-destination-link` 値に対応する `message-destination-name` と、関連付けられている `jndi-name` が含まれており、これによって JNDI ルックアップから送信先オブジェクトが解決されます。

サンプル

```
<ejb-jar>
   ...
   <assembly-descriptor>
      ...
      <message-destination>
         <message-destination-name>myAppQueue</message-destination-name>
         <jndi-name>jms/queues/TibcoQueue1</jndi-name>
      </message-destination>
      ...
   </assembly-descriptor>
</ejb-jar>
```

関連要素

親要素

- <assembly-descriptor> 要素

子要素

- <message-destination-name> 要素
- <jndi-name> 要素

<message-destination-name> 要素

```
<element name="message-destination-name" type="xsd:string"/>
```

この要素は、JMS キューやトピックなど、対象となるメッセージの送信先に割り当てられる論理名を指定します。この名前は、共通のターゲットである送信先を識別し、`message-destination-ref` または `message-driven` 要素の `message-destination-link` とともに使用して、アプリケーションにおけるメッセージ フローを表します。

サンプル

標準 EJB アプリケーション ディスクリプタ ejb-jar.xml :

```
<ejb-jar>
   <enterprise-beans>
      <session>
         ...
         <message-destination-ref>
            <message-destination-ref-name>jms/TargetQueue</message-
destination-ref-name>
            <message-destination-type>javax.jms.Queue</message-destination-
type>
            <message-destination-usage>Produces</message-destination-usage>
            <message-destination-link>myAppQueue</message-destination-link>
         </message-destination-ref>
         ...
      </session>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         ...
         <message-destination-link>myAppQueue</message-destination-link>
         ...
      </message-driven>
      ...
   </enterprise-beans>
   <assembly-descriptor>
     ...
     <message-destination>
        <message-destination-name>myAppQueue</message-destination-name>
     </message-destination>
     ...
   </assembly-descriptor>
</ejb-jar>
```

Borland EJB アプリケーション ディスクリプタ ejb-borland.xml :

```
<ejb-jar>
   <enterprise-beans>
      <session>
         ...
         <message-destination-ref>
            <message-destination-ref-name>jms/TargetQueue</message-
destination-ref-name>
            <jndi-name>jms/queues/Queue1</message-destination-type>
         </message-destination-ref>
         ...
      </session>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
               <message-driven-destination-name>jms/Q2</message-driven-
```

```
destination-name>
                <connection-factory-name>jms/xacf</connection-factory-name>
                <pool>
                   <max-size>20</max-size>
                   <init-size>1</init-size>
                </pool>
             </jms-provider-ref>
          </message-source>
          ...
       </message-driven>
       ...
    </enterprise-beans>
    <assembly-descriptor>
      ...
      <message-destination>
         <message-destination-name>myAppQueue</message-destination-name>
         <jndi-name>jms/queues/TibcoQueue</jndi-name>
      </message-destination>
      ...
    </assembly-descriptor>
</ejb-jar>
```

`message-destination-link` を介して、このアプリケーションが、名前が **jms/TargetQueue** の `message-destination-ref` に対して JNDI ルックアップを実行した場合、`message-destination` の `jndi-name` **jms/queues/TibcoQueue** が解決され、`jndi-name` **jms/queues/Queue1** とはならない点に注意してください。同様に、MDB MessageReflectorEJB の `message-destination-link` の指定を介した場合、EJB コンテナは、`message-driven-destination-name` **jms/Q2** ではなく、**jms/queues/TibcoQueue** から送信先を解決します。

関連要素

親要素

- <message-destination> 要素

子要素

- なし

<message-destination-ref> 要素

```
<complexType name="message-destination-refType">
 <sequence>
  <element name="message-destination-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string"/>
 </sequence>
</complexType>

<xsd: element name="message-destination-ref" type="borl:message-destination-
refType" minOccurs="0" maxOccurs="unbounded"/>
```

この要素は、エンタープライズ Bean のコンテキスト内の JMS キューやトピックなど、メッセージの送信先リファレンスを定義するために使用されます。各メッセージの送信先リファレンスには、Bean によって使用される `message-destination-ref-name` と、関連付けられている `jndi-name` が含まれており、これによって JNDI ルックアップから目的のオブジェクトが解決されます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
     <session>
       ...
       <message-destination-ref>
          <message-destination-ref-name>jms/StockQueue</
                    message-destination-ref-name>
          <jndi-name>jms/queues/Queue1</message-destination-type>
       </message-destination-ref>
       ...
     </session>
     ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <enterprise-beans> 要素

子要素

- <message-destination-ref-name> 要素
- <jndi-name> 要素

<message-destination-ref-name> 要素

<xsd: element name="message-destination-ref-name" type="xsd:string"/>

この要素は、JMS キューやトピックなど、メッセージの送信先リファレンスにアクセスするために、エンタープライズ Bean が使用する論理名を指定します。この名前は、アプリケーション コンポーネントの java:comp/env への JNDI 相対名です。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         ...
         <message-destination-ref>
            <message-destination-ref-name>jms/StockQueue
                        </message-destination-ref-name>
            <jndi-name>jms/queues/Queue1</message-destination-type>
         </message-destination-ref>
         ...
      </session>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <message-destination-ref> 要素

子要素

- なし

<message-driven-destination-name> 要素

<xsd: element name="message-driven-destination-name" type="xsd:string"/>

Bean がサブスクライブする個別のキューまたはトピックの JNDI 名を指定します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
              <message-driven-destination-name>serial://jms/q
                              </message-driven-destination-name>
               <connection-factory-name>serial://jms/xacf
                              </connection-factory-name>
               <pool>
                  <max-size>20</max-size>
                  <init-size>1</init-size>
               </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <jms-provider-ref> 要素

子要素

- なし

<message-driven> 要素

```
<xsd:element name="message-driven">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="ejb-name" type="xsd:string"/>
   <xsd:element name="message-source">
    <xsd:complexType>
     <xsd:choice>
      <xsd:element name="resource-adapter-ref" type="borl:resource-adapter-refType"/>
      <xsd:element name="jms-provider-ref" type="borl:jms-provider-refType"/>
     </xsd:choice>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
   <xsd:element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
   <xsd:element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
   <xsd:element name="resource-env-ref" type="borl:resource-env-refMdbType" minOccurs="0"
maxOccurs="unbounded"/>
   <xsd:element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
   <xsd:element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

アーカイブにデプロイされるメッセージ駆動型 Bean を記述します。この要素の子ノードを使用して、Bean のさまざまなインターフェイスとリファレンスを指定できます。Bean の接続先のキューやトピック、および接続に使用する接続ファクトリに関するデータを提供することもできます。プールでの Bean の動作を指定することもできます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
               <message-driven-destination-name>serial://jms/q</message-driven-
destination-name>
               <connection-factory-name>serial://jms/xacf</connection-factory-
name>
               <pool>
                  <max-size>20</max-size>
                  <init-size>1</init-size>
               </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <enterprise-beans> 要素

子要素

- <ejb-name> 要素
- <message-source> 要素
- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination-ref> 要素
- <property> 要素

<message-source> 要素

```
<xsd:element name="message-source">
 <xsd:complexType>
  <xsd:choice>
   <xsd:element name="resource-adapter-ref" type="borl:
        resource-adapter-refType"/>
   <xsd:element name="jms-provider-ref" type="borl:jms-provider-refType"/>
  </xsd:choice>
 </xsd:complexType>
</xsd:element>
```

message-source 要素は、 デプロイされた MDB のアクティベーションが、 EJB 2.0 と EJB 2.1 のどちらの仕様にしたがって実行されるかを、 EJB コンテナに対して指示します。 EJB 2.0 の場合には jms-provide-ref の子要素を持ち、 EJB 2.1 の場合には、 子要素 resource-adapter-ref で指定されたリソース アダプタ経由で、 メッセージが配信されます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
               <message-driven-destination-name>serial://jms/q
                                    </message-driven-destination-name>
              <connection-factory-name>serial://jms/xacf</connection-factory-
name>
               <pool>
                  <max-size>20</max-size>
                  <init-size>1</init-size>
               </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <message-driven> 要素

子要素

- <resource-adapter-ref> 要素
- <jms-provider-ref> 要素

<method-name> 要素

```
<xsd:element name="method-name" type="xsd:string"/>
```

この method 要素は、標準ディスクリプタ内に存在する Entity Bean クエリのうち、Borland ディスクリプタ内でなんらかの変更が加えられているものについて、そのメソッド シグニチャの部分を識別します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         ...
         <query>
            <query-method>
              <method-name>findByStudent</method-name>
               <method-params>
                    <method-param>java.lang.String</method-param>
               </method-params>
            </query-method>
            <user-sql>SELECT course_dept, course_number FROM Enrollment
                        i WHERE i.student=?1</user-sql>
            <load-state>True</load-state>
         </query>
      </entity>
      ...
   </enterprise-beans>
   ...
</ejb-jar>
```

関連要素

親要素

- EJB モジュール：ejb-borland.xml

子要素

- なし

<method-param> 要素

```
<xsd:element name="method-param" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
```

この method-param 要素は、標準ディスクリプタ内に存在する Entity Bean クエリのうち、Borland ディスクリプタ内でなんらかの変更が加えられているものについて、そのメソッド シグニチャ内のパラメータを識別します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         ...
         <query>
            <query-method>
               <method-name>findByStudent</method-name>
                <method-params>
                     <method-param>java.lang.String</method-param>
                </method-params>
            </query-method>
            <user-sql>SELECT course_dept, course_number FROM Enrollment
                        i WHERE i.student=?1</user-sql>
            <load-state>True</load-state>
         </query>
      </entity>
      ...
   </enterprise-beans>
   ...
</ejb-jar>
```

関連要素

親要素

- EJB モジュール：ejb-borland.xml

子要素

- なし

<method-params> 要素

```
<xsd:element name="method-params">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="method-param" type="xsd:string" minOccurs=
          "0" maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この method-params 要素は、標準ディスクリプタ内に存在する Entity Bean クエリ要素のうち、Borland ディスクリプタ内でなんらかの変更が加えられているものについて、そのメソッド シグニチャ内のすべてのパラメータを識別します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         ...
         <query>
            <query-method>
               <method-name>findByStudent</method-name>
                <method-params>
                   <method-param>java.lang.String</method-param>
               </method-params>
            </query-method>
            <user-sql>SELECT course_dept, course_number FROM Enrollment
                     i WHERE i.student=?1</user-sql>
            <load-state>True</load-state>
         </query>
      </entity>
      ...
   </enterprise-beans>
   ...
</ejb-jar>
```

関連要素

親要素

- EJB モジュール：ejb-borland.xml

子要素

- EJB モジュール：ejb-borland.xml

<method-signature> 要素

<xsd: element name="method-signature" type="xsd:string"/>

データベース クエリの実行に使用されるメソッド。エンティティ Bean に示されるとおりです。

サンプル

<method-signature>findByStudent(Student s)</method-signature>

関連要素

親要素

- <finder> 要素

子要素

- なし

<password> 要素

<xsd: element name="password" type="xsd:string" minOccurs="0"/>

JDBC 1.x 専用。定義されるデータソースにアクセスするためのパスワード。

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
        <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver</driver-
class-name>
     </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <datasource> 要素

子要素

- なし

<pool> 要素

```
<xsd:element name="pool" minOccurs="0">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="max-size" type="xsd:string" minOccurs="0"/>
      <xsd:element name="init-size" type="xsd:string" minOccurs="0"/>
      <xsd:element name="wait-timeout" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

javax.jms.MessageListener を使用して JMS プロバイダから直接メッセージを消費するように設定されている場合に、MDB のリソース プールのプロパティを指定できる子ノードを含みます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
   <message-driven>
      <ejb-name>MessageReflectorEJB</ejb-name>
      <message-source>
         <jms-provider-ref>
         <message-driven-destination-name>serial://jms/q</message-driven-
destination-name>
         <connection-factory-name>serial://jms/xacf</connection-factory-name>
         <pool>
            <max-size>20</max-size>
            <init-size>1</init-size>
         </pool>
         </jms-provider-ref>
      </message-source>
      <resource-ref>
         <res-ref-name>jms/ConnectionFactory</res-ref-name>
         <jndi-name>serial://jms/xacf</jndi-name>
      </resource-ref>
      <resource-env-ref>
         <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
         <jndi-name>serial://jms/t</jndi-name>
      </resource-env-ref>
      </message-driven>
     ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <jms-provider-ref> 要素

子要素

- <max-size> 要素
- <init-size> 要素
- <wait-timeout> 要素

<property> 要素

```
<xsd: element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="propertyType">
 <xsd:sequence>
  <xsd:element name="prop-name" type="xsd:string"/>
  <xsd:element name="prop-type" type="xsd:string" minOccurs="0"/>
  <xsd:element name="prop-value" type="xsd:string"/>
 </xsd:sequence>
</xsd:complexType>
```

この要素は、アーカイブまたはそのコンポーネントに含まれるか、それらから参照されるさまざまなリソースのプロパティ値を指定するために使用されます。各 `property` エントリは、対応するサブ要素を使用して、プロパティの名前、型、および値を指定します。

サンプル

```
<property>
   <prop-name>vbroker.security.disable</prop-name>
   <prop-type>security</prop-type>
   <prop-value>false</prop-value>
</property>
```

関連要素

親要素

- <admin-object> 要素
- <cmp-field> 要素
- <cmr-field> 要素
- <session> 要素
- <entity> 要素
- <message-driven> 要素
- <datasource> 要素
- <column-properties> 要素
- <table-properties> 要素

子要素

- <prop-name> 要素
- <prop-type> 要素
- <prop-value> 要素

<prop-name> 要素

```
<xsd: element name="prop-name" type="xsd:string"/>
```

設定するプロパティの名前を指定します。

サンプル

```
<prop-name>vbroker.security.disable</prop-name>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-type> 要素

```
<xsd: element name="prop-type" type="xsd:string" minOccurs="0"/>
```

設定するプロパティの型を指定します。

サンプル

```
<prop-type>security</prop-type>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-value> 要素

```
<xsd: element name="prop-value" type="xsd:string"/>
```

設定するプロパティの値を指定します。

サンプル

```
<prop-value>false</prop-value>
```

関連要素

親要素

- <property> 要素
- <jdbc-property> 要素

子要素

- なし

<query> 要素

```
<xsd:element name="query" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="query-method">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="method-name" type="xsd:string"/>
      <xsd:element name="method-params">
       <xsd:complexType>
        <xsd:sequence>
         <xsd:element name="method-param" type="xsd:string" minOccurs=
                  "0" maxOccurs="unbounded"/>
        </xsd:sequence>
       </xsd:complexType>
      </xsd:element>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="user-sql" type="xsd:string" minOccurs="0"/>
   <xsd:element name="load-state" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この要素は、標準ディスクリプタ内に存在するエンティティ Bean クエリのうち、load-state や user-sql 要素コンテンツなどに Borland ディスクリプタ内でなんらかの変更が加えられているものを識別します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         ...
         <query>
            <query-method>
               <method-name>findByStudent</method-name>
               <method-params>
                  <method-param>java.lang.String</method-param>
               </method-params>
            </query-method>
            <user-sql>SELECT course_dept, course_number FROM Enrollment
                     i WHERE i.student=?1</user-sql>
            <load-state>True</load-state>
         </query>
      </entity>
      ...
   </enterprise-beans>
   ...
</ejb-jar>
```

関連要素

親要素

- <entity> 要素

子要素

- <query-method> 要素
- <user-sql> 要素
- <load-state> 要素

<query-method> 要素

```
<xsd:element name="query-method">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="method-name" type="xsd:string"/>
   <xsd:element name="method-params">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="method-param" type="xsd:string" minOccurs=
                "0" maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   <xsd:/element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この query-method 要素は、標準ディスクリプタ内に存在する Entity Bean クエリのうち、Borland ディスクリプタ内でなんらかの変更が加えられているものを識別する重要な情報を提供します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         ...
         <query>
            <query-method>
               <method-name>findByStudent</method-name>
               <method-params>
                  <method-param>java.lang.String</method-param>
               </method-params>
            </query-method>
             <user-sql>SELECT course_dept, course_number FROM Enrollment
                        i WHERE i.student=?1</user-sql>
             <load-state>True</load-state>
         </query>
      </entity>
      ...
   </enterprise-beans>
   ...
</ejb-jar>
```

関連要素

親要素

- <query> 要素

子要素

- <method-name> 要素
- <method-params> 要素

<relationship-role-source> 要素

```
<xsd:element name="relationship-role-source">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="ejb-name" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

別のエンティティ Bean とコンテナ管理の関係にあるエンティティ Bean の名前を指定します。

サンプル

```
<relationship-role-source>
   <ejb-name>Customer</ejb-name>
</relationship-role-source>
```

関連要素

親要素

- <ejb-relationship-role> 要素

子要素

- <ejb-name> 要素

<relationships> 要素

```
<xsd:element name="relationships" minOccurs="0">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ejb-relation" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="ejb-relationship-role" type=
                        "borl:ejb-relationship-roleType"/>
            <xsd:element name="ejb-relationship-role" type=
                        "borl:ejb-relationship-roleType"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

テーブル関係を指定するには、 `<relationships>` 要素を使用します。 `<relationships>` 要素内で、 ロールのソース（エンティティ Bean）を保持する `<ejb-relationship-role>` 要素と、 関係を保持する `<cmr-field>` 要素を定義します。 その後、 ディスクリプタは <table-ref> 要素を使用して 2 つのテーブル `<left-table>` と `<right-table>` の間の関係を指定します。 次の多重度（カーディナリティ）を指定できます。

- 方向ごとに <ejb-relationship-role> を定義する必要があります。双方向関係の場合、相互に関係のある Bean ごとに <ejb-relationship-role> を定義してください。
- 関係 1 つにつき、使用できる <table-ref> 要素は 1 つだけです。

多対多の関係を定義する場合、 CMP エンジンに、左右テーブルの関係をモデル化する、クロステーブルを作成させなければなりません。これを行う際に <cross-table> 要素を使用します。

サンプル

```
<ejb-jar>
  ...
  <relationships>
    <ejb-relation>
      <ejb-relationship-role>
        <relationship-role-source>
          <ejb-name>Customer</ejb-name>
        </relationship-role-source>
        <cmr-field>
          <cmr-field-name>specialInformation</cmr-field-name>
          <table-ref>
            <left-table>
              <table-name>CUSTOMER</table-name>
              <column-list>
                <column-name>CUSTOMER_NO</column-name>
              </column-list>
            </left-table>
            <right-table>
              <table-name>SPECIAL_INFO</table-name>
              <column-list>
                <column-name>CUSTOMER_NO</column-name>
              </column-list>
            </right-table>
          </table-ref>
        </cmr-field>
      </ejb-relationship-role>
      <ejb-relationship-role>
        <relationship-role-source>
          <ejb-name>SpecialInfo</ejb-name>
        </relationship-role-source>
      </ejb-relationship-role>
    </ejb-relation>
  </relationships>
  ...
</ejb-jar>
```

関連要素

親要素

- <ejb-jar> 要素

子要素

- <ejb-relation> 要素

<resource-env-ref-name> 要素

<xsd: element name="resource-env-ref-name" type="xsd:string"/>

この要素は、Bean がリソース環境リファレンスにアクセスするために使用する名前を提供します。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         ...
         <resource-env-ref>
            <resource-env-ref-name>jms/targetQueue</resource-env-ref-name>
            <jndi-name>jms/Tibco/Queue1</jndi-name>
         </resource-env-ref>
         ...
      </session>
      <message-driven>
         ...
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>jms/Tibco/Topic1</jndi-name>
         </resource-env-ref>
         ...
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <resource-env-ref> 要素

子要素

- なし

<resource-adapter-ref> 要素

```
<xsd:complexType name="resource-adapter-refType">
 <xsd:sequence>
  <xsd:element name="instance-name" type="xsd:string"/>
 </xsd:sequence>
</xsd:complexType>

<element name="resource-adapter-ref" type="borl:resource-adapter-refType"/>
```

resource-adapter-ref 要素は、デプロイされた MDB のアクティベーションが、EJB 2.1 の仕様にしたがって実行されることを、EJB コンテナに対して指示します。この場合、メッセージは、instance-name 子要素によって指定されるリソース アダプタ経由で配信されます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <resource-adapter-ref>
               <instance-name>mailAdapter</instance-name>
            </resource-adapter-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <message-source> 要素

子要素

- <instance-name> 要素

<resource-env-ref> 要素

```
<xsd: resource-env-ref" type="borl:resource-env-refType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="resource-env-refType">
 <xsd:sequence>
  <xsd:element name="resource-env-ref-name" type="xsd:string"/>
  <xsd:element name="jndi-name" type="xsd:string"/>
 </xsd:sequence>
</xsd:complexType>

<xsd: resource-env-ref" type="borl:resource-env-refMdbType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="resource-env-refMdbType">
 <xsd:sequence>
  <xsd:element name="resource-env-ref-name" type="xsd:string"/>
  <xsd:choice>
   <xsd:element name="admin-object" type="borl:admin-objectType"/>
   <xsd:element name="jndi-name" type="xsd:string"/>
  </xsd:choice>
 </xsd:sequence>
</xsd:complexType>
```

この要素は、Bean によって使用されるリソース環境リファレンスを JNDI 名、またはリソース アダプタの管理オブジェクトのいずれかにマップするために使用されます。この要素には、2 つのフレーバーがあります。セッションおよびエンティティ Bean の場合は、JNDI オブジェクトを解決するためだけに使用されます。メッセージ駆動型 Bean の場合は、管理オブジェクトまたは JNDI オブジェクトのいずれかを解決するために使用されます。これは、この Bean がリソース アダプタからメッセージを消費するように設定されているか、または message-source 要素に指定されているように JMS プロバイダから直接メッセージを消費するように設定されているかによって異なります。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
               <message-driven-destination-name>serial://jms/q
                              </message-driven-destination-name>
               <connection-factory-name>serial://jms/xacf
                              </connection-factory-name>
               <pool>
                  <max-size>20</max-size>
                  <init-size>1</init-size>
               </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
           <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
           <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <session> 要素
- <entity> 要素
- <message-driven> 要素

子要素

- <resource-env-ref-name> 要素
- <admin-object> 要素
- <jndi-name> 要素

<resource-ref> 要素

```
<xsd: element name="resource-ref" type="borl:resource-refType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="resource-refType">
 <xsd:sequence>
  <xsd:element name="res-ref-name" type="xsd:string"/>
  <xsd:element name="jndi-name" type="xsd:string"/>
  <xsd:element name="cmp-resource" type="xsd:string" minOccurs="0"/>
 </xsd:sequence>
</xsd:complexType>
```

この要素は、 Bean によって使用されるリソース リファレンスを定義するために使用されます。各リソース リファレンスには、 クライアント アプリケーションによって使用される `res-ref-name` と、それに関連付けられた `jndi-name` （該当する場合）が含まれています。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         ...
         <resource-ref>
            <res-ref-name>jdbc/CheckingDataSource</res-ref-name>
            <jndi-name>jdbc/datasources/OracleDataSource</jndi-name>
         </resource-ref>
         ...
      </session>
      <message-driven>
         <ejb-name>MessageReflectorEJB</ejb-name>
         <message-source>
            <jms-provider-ref>
               <message-driven-destination-name>serial://jms/q</message-driven-
destination-name>
               <connection-factory-name>serial://jms/xacf</connection-factory-
name>
               <pool>
                  <max-size>20</max-size>
                  <init-size>1</init-size>
               </pool>
            </jms-provider-ref>
         </message-source>
         <resource-ref>
            <res-ref-name>jms/ConnectionFactory</res-ref-name>
            <jndi-name>serial://jms/xacf</jndi-name>
         </resource-ref>
         <resource-env-ref>
            <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
            <jndi-name>serial://jms/t</jndi-name>
         </resource-env-ref>
      </message-driven>
      ...
   </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <session> 要素

- <entity> 要素
- <message-driven> 要素

子要素

- <res-ref-name> 要素
- <jndi-name> 要素
- <cmp-resource> 要素

<res-ref-name> 要素

<xsd: element name="res-ref-name" type="xsd:string"/>

この要素は、Bean がリソース リファレンスにアクセスするために使用する名前を提供します。

サンプル

<res-ref-name>jdbc/CheckingDataSource</res-ref-name>

関連要素

親要素

- <resource-ref> 要素

子要素

- なし

<right-table> 要素

```
<xsd:element name="right-table">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="table-name" type="xsd:string"/>
  <xsd:element name="column-list" type="borl:column-listType"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

`<cmp2-info>` を指定する場合、この要素は、1 つの列を共有する 2 つのテーブルの一方を定義します。一方がもう一方の外部キーになります。

この要素を使って `<relationships>` を記述する場合、`<left-table>` が関係の元、`<right-table>` が関係の先となります。つまり、関係の方向は左から右です。双方向の関係を定義する場合は、各方向ごとに 1 つ、つまり 1 つの関係に対して 2 つの `<table-ref>` 要素を作成する必要があります。多対多の関係では、`<right-table>` は、共通部分を定義する `<cross-table>` を作成するために使用される 2 つのテーブルの一方を作成します。

サンプル

```
<right-table>
    <table-name>CUSTOMER</table-name>
    <column-list>CUSTOMER_NO</column-list>
</right-table>
```

関連要素

親要素

- <table-ref> 要素
- <table-name> 要素
- <column-list> 要素

<role-name> 要素

```
<xsd: element name="role-name" type="xsd:string"/>
```

アーカイブ内のモジュールによって使用されるセキュリティ ロールのロール名。

サンプル

```
<role-name>administrator</role-name>
```

関連要素

親要素

- <security-role> 要素

子要素

- なし

<security-role> 要素

```
<xsd:element name="security-role" minOccurs="0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="role-name" type="xsd:string"/>
      <xsd:element name="deployment-role" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

アーカイブ内のモジュールで使用されるセキュリティ ロールとデプロイメント ロール（該当する場合）の名前を指定します。

サンプル

```
<security-role>
   <role-name>administrator</role-name>
   <deployment-role>administrator</deployment-role>
</security-role>
```

関連要素

親要素

- <assembly-descriptor> 要素

子要素

- <role-name> 要素
- <deployment-role> 要素

<session> 要素

```
<xsd:element name="session">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="ejb-name" type="xsd:string"/>
   <xsd:element name="bean-home-name" type="xsd:string" minOccurs="0"/>
   <xsd:element name="bean-local-home-name" type="xsd:string" minOccurs="0"/>
   <xsd:element name="timeout" type="xsd:string" minOccurs="0"/>
   <xsd:element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
   <xsd:element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
   <xsd:element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
   <xsd:element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/
>
   <xsd:element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
   <xsd:element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

`<session>` 要素は、アーカイブに含まれるセッション Bean に関する情報を提供します。この要素の子ノードを使用して、Bean のさまざまなインターフェイスとリファレンスを指定できます。セッション Bean に固有の属性（`timeout` など）のほか、個別のセッション Bean に固有の汎用プロパティも提供できます。

サンプル

```
<session>
    <ejb-name>UniqueIdGeneratorEJB</ejb-name>
    <bean-local-home-name>ejb/local/petstore/uidgen/UniqueIdGenerator
        </bean-local-home-name>
    <timeout>0</timeout>
    <ejb-local-ref>
       <ejb-ref-name>ejb/local/Counter</ejb-ref-name>
    </ejb-local-ref>
</session>
```

関連要素

親要素

- <enterprise-beans> 要素

子要素

- <ejb-name> 要素
- <bean-home-name> 要素
- <bean-local-home-name> 要素
- <timeout> 要素
- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination-ref> 要素
- <property> 要素

<table> 要素

```
<xsd: element name="table" type="xsd:string" minOccurs="0"/>
```

CMP 1.x エンティティ Bean が自分のフィールドにデータを格納するために使用するデータベース テーブルの名前を指定します。

サンプル

```
<table>Course</table>
```

関連要素

親要素

- <database-map> 要素

子要素

- なし

<table-name> 要素

```
<xsd: element name="table-name" type="xsd:string"/>
```

エンティティ マッピングまたはプロパティ設定のためのテーブルの名前を指定します。

サンプル

```
<table-name>CUSTOMER</table-name>
```

関連要素

親要素

- <left-table> 要素
- <right-table> 要素
- <cross-table> 要素
- <table-properties> 要素
- <cmp2-info> 要素

子要素

- なし

<table-properties> 要素

```
<xsd:element name="table-properties" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="table-name" type="xsd:string"/>
   <xsd:element name="column-properties" minOccurs="0"
maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="column-name" type="xsd:string"/>
      <xsd:element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

この要素を使用して、エンティティのデータベース リソースに関する詳細情報を提供します。その子ノードを使用して、テーブルの名前と、テーブルに関連付けられたプロパティを指定します（`create_tables` など）。`<column-properties>` 子ノードを使用して、データベース列に固有のプロパティを指定することもできます。

サンプル

```
<table-properties>
 <table-name>CUSTOMER</table-name>
  <property>
   <prop-name>create-tables</prop-name>
   <prop-value>True</prop-value>
  </property>
</table-properties>
```

関連要素

親要素

- <ejb-jar> 要素

子要素

- <table-name> 要素
- <column-properties> 要素
- <property> 要素

<table-ref> 要素

```
<xsd: element name="table-ref" type="borl:table-refType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd: element name="table-ref" type="borl:table-refType"/>

<xsd:complexType name="table-refType">
 <xsd:sequence>
  <xsd:element name="left-table">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="table-name" type="xsd:string"/>
     <xsd:element name="column-list" type="borl:column-listType"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="cross-table" minOccurs="0" maxOccurs="unbounded">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="table-name" type="xsd:string"/>
     <xsd:element name="column-list" type="borl:column-listType"/>
     <xsd:element name="column-list" type="borl:column-listType"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
  <xsd:element name="right-table">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="table-name" type="xsd:string"/>
     <xsd:element name="column-list" type="borl:column-listType"/>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>
```

<cmp2-info> を指定する場合は、複数のテーブルに永続化される情報を含むエンティティを設定できます。これらのテーブルは、最低でも 1 つ、リンクされたテーブルでの外部キーを表す列でリンクする必要があります。この関係は、<table-ref> 要素を使用して記述できます。その子ノード <left-table> や <right-table> を使用して、関係を持つテーブル群と、それらが共有する列（複数可）を指定してください。

この要素は、<relationships> 要素を使用して、テーブル間の関係を記述するためにも使用されます。この場合、ディスクリプタでは、<table-ref> を使用して、<left-table> と <right-table> の間の関係を指定します。その際、次の原則を守る必要があります：

- 一方向につき <ejb-relationship-role> 要素 1 つを定義する必要があります。双方向関係の場合、お互いに参照している Bean それぞれに対して、<ejb-relationship-role> を定義する必要があります。
- 関係 1 つにつき、使用できる <table-ref> 要素は 1 つだけです。

多対多の関係を定義する場合、CMP エンジンに、左右テーブルの関係をモデル化する、クロステーブルを作成させなければなりません。これを行う際に、<cross-table> 要素を使用します。

サンプル

```
<table-ref>
  <left-table>
   <table-name>LINE_ITEM</table-name>
   <column-list>
    <column-name>LINE</column-name>
   </column-list>
  </left-table>
  <right-table>
   <table-name>QUANTITY</table-name>
   <column-list>
    <column-name>LINE</column-name>
   </column-list>
  </right-table>
 </table-ref>
```

関連要素

親要素

- <cmp2-info> 要素
- <cmr-field> 要素

子要素

- <left-table> 要素
- <right-table> 要素
- <cross-table> 要素

<timeout> 要素

<xsd: element name="timeout" type="xsd:string" minOccurs="0"/>

セッション Bean が呼び出しを待機してタイムアウトになるまでの時間を秒単位で指定します。デフォルトは 0 秒です。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <session>
         <ejb-name>clerk</ejb-name>
         <bean-home-name>insurance/remote/clerk</bean-home-name>
         <timeout>5</timeout>
         <ejb-local-ref>
            <ejb-ref-name>ejb/insurance/claim</ejb-ref-name>
         </ejb-local-ref>
         <resource-ref>
            <res-ref-name>jms/insurance/ConnectionFactory</res-ref-name>
            <jndi-name>jms/xacf</jndi-name>
         </resource-ref>
      </session>
      ...
    </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <session> 要素

子要素

- なし

<url> 要素

<xsd: element name="url" type="xsd:string"/>

JDBC 1.x 専用。定義されるデータソースの URL。

関連要素

親要素

- <datasource> 要素

子要素

- なし

<username> 要素

<xsd: element name="username" type="xsd:string" minOccurs="0"/>

JDBC 1.x 専用。定義されるデータソースにアクセスするためのユーザー名。

サンプル

```
<ejb-jar>
   ...
   <datasource-definitions>
     <datasource>
        <jndi-name>datasources/ComplexDataSource</jndi-name>
        <url>jdbc:borland:dslocal:ejbcontainer</url>
        <username>sysdba</username>
        <password>masterkey</password>
        <driver-class-name>com.borland.datastore.jdbc.DataStoreDriver
                </driver-class-name>
     </datasource>
   </datasource-definitions>
   ...
</ejb-jar>
```

関連要素

親要素

- <datasource> 要素

子要素

- なし

<user-sql> 要素

<xsd: element name="user-sql" type="xsd:string" minOccurs="0"/>

この要素を使用すると、SQL の仕様をクエリ メソッドと関連付けることができます。

サンプル

```
<ejb-jar>
   <enterprise-beans>
      <entity>
         ...
         <query>
            <query-method>
               <method-name>findByStudent</method-name>
               <method-params>
                  <method-param>java.lang.String</method-param>
               </method-params>
            </query-method>
            <user-sql>SELECT course_dept, course_number FROM Enrollment i
WHERE i.student=?1</user-sql>
            <load-state>False</load-state>
         </query>
      </entity>
      ...
   </enterprise-beans>
   ...
</ejb-jar>
```

関連要素

親要素

- <query> 要素

子要素

- なし

<wait-timeout> 要素

<xsd: element name="wait-timeout" type="xsd:string" minOccurs="0"/>

`max-size` 要素がすでにオープンになっている場合は、MDB プールの要素が解放されるまで待つ時間を秒数で表します。`max-size` プロパティを使用しており、プールがこれ以上接続を使用できない場合、接続を検索するスレッドは、待ち時間が無制限に設定されている（0 秒に設定）と、その接続が使用できるようになるまで待機します。必要に応じて、wait-timeout 時間を設定できます。これは `ejb.mdb.wait_timeout` プロパティと同じです。

サンプル

```
<ejb-jar>
  <enterprise-beans>
    <message-driven>
      <ejb-name>MessageReflectorEJB</ejb-name>
      <message-source>
        <jms-provider-ref>
          <message-driven-destination-name>serial://jms/q
                    </message-driven-destination-name>
          <connection-factory-name>serial://jms/xacf
                    </connection-factory-name>
          <pool>
            <max-size>20</max-size>
            <init-size>1</init-size>
            <wait-timeout>20</wait-timeout>
          </pool>
        </jms-provider-ref>
      </message-source>
      <resource-ref>
        <res-ref-name>jms/ConnectionFactory</res-ref-name>
        <jndi-name>serial://jms/xacf</jndi-name>
      </resource-ref>
      <resource-env-ref>
        <resource-env-ref-name>jms/ReplyTopic</resource-env-ref-name>
        <jndi-name>serial://jms/t</jndi-name>
      </resource-env-ref>
    </message-driven>
    ...
  </enterprise-beans>
</ejb-jar>
```

関連要素

親要素

- <pool> 要素

子要素

- なし

<where-clause> 要素

```
<xsd: element name="where-clause" type="xsd:string"/>
```

where 節は、取得するレコードの範囲を限定する場合に必要な select 文の一部です。where 節の構文はかなり複雑になることがあり、EJB コンテナがこの節を正しく生成できるように、XML デプロイメント ディスクリプタ ファイルでは、一定の規則にしたがう必要があります。

まず、必ずしも <where-clause> で「where」リテラルを使用する必要はありません。このリテラルのない where 節を生成し、where 節記入はコンテナに任せることができます。ただし、コンテナは、<where-clause> 内が空文字列でない場合にのみ、これを行います。空文字列は空のままになります。たとえば、次のどちらの方法でも同じ where 節を定義できます。

```
<where-clause> where a = b </where-clause>
```

または

```
<where-clause> a = b </where-clause>
```

コンテナは、`a = b` を同義の where 節、`where a = b` に変換します。しかし、`<where-clause> "" </where-clause>` と定義されている空文字列の場合には、変換は行われません。

パラメータ置換は、where 節の重要な部分です。Borland EJB コンテナは、標準の SQL 置換接頭辞であるコロン（:）を見つけると、パラメータ置換を行います。置換される各パラメータは、XML デプロイメント ディスクリプタにある検索メソッド定義中のパラメータの名前に対応します。

コンテナは、複合パラメータもサポートします。複合パラメータは、テーブル名の後に、そのテーブル内の列が付きます。複合パラメータには、標準のドット（.）構文を使用します。この構文では、テーブル名と列名をドットで区切ります。複合パラメータでも、前にコロンを付けます。

エンティティ Bean は、検索メソッドのパラメータとしても使用できます。エンティティ Bean は複合型として使用できます。その場合、SQL クエリに渡すエンティティ Bean リファレンスのどのフィールドを使用するのかを CMP エンジンに指示する必要があります。複合型としてエンティティ Bean を使用しなければ、コンテナは、where 節にその Bean の主キーを代入します。

サンプル

```
<where-clause>SELECT course_dept, course_number FROM
  Enrollment WHERE student  = :s[ejb/Student]</where-clause>
```

関連要素

親要素

- <finder> 要素

子要素

- なし

第 6 章

Web モジュール：web-borland.xml

XSD: web-app_2_4-borland.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://support.borland.com/appserver/xml/ns/j2ee" xmlns:xsd="http://www.w3.org/2001/
XMLSchema" xmlns:borl="http://support.borland.com/appserver/xml/ns/j2ee" xmlns="http://www.w3.org/2001/
XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.4">
 <!-- Start definition of ComplexTypes -->
 <complexType name="ejb-refType">
  <sequence>
   <element name="ejb-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
 <complexType name="ejb-local-refType">
  <sequence>
   <element name="ejb-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
 <complexType name="resource-refType">
  <sequence>
   <element name="res-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string"/>
  </sequence>
 </complexType>
 <complexType name="resource-env-refType">
  <sequence>
   <element name="resource-env-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string"/>
  </sequence>
 </complexType>
 <complexType name="message-destination-refType">
  <sequence>
   <element name="message-destination-ref-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string"/>
  </sequence>
 </complexType>
 <complexType name="propertyType">
```

```
  <sequence>
   <element name="prop-name" type="xsd:string"/>
   <element name="prop-type" type="xsd:string" minOccurs="0"/>
   <element name="prop-value" type="xsd:string"/>
  </sequence>
 </complexType>
 <complexType name="web-deploy-pathType">
  <sequence>
   <element name="service" type="xsd:string"/>
   <element name="engine" type="xsd:string"/>
   <element name="host" type="xsd:string"/>
  </sequence>
 </complexType>
 <complexType name="security-roleType">
  <sequence>
   <element name="role-name" type="xsd:string"/>
   <element name="deployment-role" type="xsd:string" minOccurs="0"/>
  </sequence>
 </complexType>
 <complexType name="message-destinationType">
  <sequence>
   <element name="message-destination-name" type="xsd:string"/>
   <element name="jndi-name" type="xsd:string"/>
  </sequence>
 </complexType>
 <element name="web-app">
  <complexType>
   <sequence>
    <element name="context-root" type="xsd:string" minOccurs="0"/>
    <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
maxOccurs="unbounded"/>
    <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="web-deploy-path" type="borl:web-deploy-pathType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="authorization-domain" type="xsd:string" minOccurs="0"/>
    <element name="security-role" type="borl:security-roleType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="message-destination" type="borl:message-destinationType" minOccurs="0"
maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
</schema>
```

<authorization-domain> 要素

```
<element name="authorization-domain" type="xsd:string" minOccurs="0"/>
```

アプリケーションが属する認証ドメインの名前。

サンプル

```
<authorization-domain>GroupJ</authorization-domain>
```

関連要素

親要素

- <web-app> 要素

子要素

- なし

<context-root> 要素

```
<element name="context-root" type="xsd:string" minOccurs="0"/>
```

通常、Web アプリケーションの名前は WAR の名前と同じです（`.war` 拡張子は除く）。`<context-root>` 構成要素を利用すると、アプリケーションの名前を任意の名前に変更することができます。

サンプル

```
<context-root>alienWare</context-root>
```

関連要素

親要素

- <web-app> 要素

子要素

- なし

<deployment-role> 要素

```
<element name="deployment-role" type="xsd:string" minOccurs="0"/>
```

Web アプリケーションを実行する BAS ロールのロール名。

サンプル

```
<deployment-role>administrator</deployment-role>
```

関連要素

- <security-role> 要素

<ejb-local-ref> 要素

```
<element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0"
maxOccurs="unbounded"/>

<complexType name="ejb-local-refType">
 <sequence>
  <element name="ejb-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string" minOccurs="0"/>
 </sequence>
</complexType>
```

この要素は、Web アプリケーションによって使用される EJB リファレンスを定義するために使用されます。各 EJB リファレンスには、アプリケーションによって使用される `message-destination-ref-name` と、それに関連付けられた `jndi-name` が含まれています。

サンプル

```
<ejb-ref>
   <ejb-ref-name>ejb/Sort</ejb-ref-name>
   <jndi-name>sort</jndi-name>
</ejb-ref>
```

関連要素

親要素

- <web-app> 要素

子要素

- <ejb-ref-name> 要素
- <jndi-name> 要素

<ejb-name> 要素

```
<element name="ejb-name" type="xsd:string"/>
```

`<ejb-name>` 要素を使用して、定義するエンタープライズ JavaBeans の名前を指定します。この要素は、`ejb-jar.xml` 内の同じ要素と同様に、リモートからの Bean のルックアップに使用する名前を指定します。

サンプル

```
<ejb-name>clerk</ejb-name>
```

関連要素

- <ejb-ref> 要素

<ejb-ref> 要素

```
<element name="ejb-ref" type="borl:ejb-refType" minOccurs="0"
maxOccurs="unbounded"/>

<complexType name="ejb-refType">
 <sequence>
  <element name="ejb-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string" minOccurs="0"/>
 </sequence>
</complexType>
```

この要素は、Web アプリケーションによって使用される EJB リファレンスを定義するために使用されます。各 EJB リファレンスには、アプリケーションによって使用される `ejb-ref-name` と、それに関連付けられた `jndi-name` が含まれています。

サンプル

```
<ejb-ref>
   <ejb-ref-name>ejb/Sort</ejb-ref-name>
   <jndi-name>sort</jndi-name>
</ejb-ref>
```

関連要素

親要素

- <web-app> 要素

子要素

- <ejb-ref-name> 要素
- <jndi-name> 要素

<ejb-ref-name> 要素

```
<element name="ejb-ref-name" type="xsd:string"/>
```

この要素は、Web アプリケーションによってリソース リファレンスとして使用される EJB の名前を提供します。

サンプル

```
<ejb-ref-name>ejb/Sort</ejb-ref-name>
```

関連要素

親要素

- <ejb-ref> 要素
- <ejb-local-ref> 要素

子要素

- なし

<engine> 要素

```
<element name="engine" type="xsd:string"/>
```

エンジン名を設定します。これは、Tomcat の `server.xml` ファイルで定義されたエンジンに対応している必要があります。

サンプル

```
<engine>cyrpi</engine>
```

関連要素

親要素

- <web-deploy-path> 要素

子要素

- なし

<host> 要素

```
<element name="host" type="xsd:string"/>
```

ホスト名を設定します。これは、Tomcat の `server.xml` ファイルで定義されたホストに対応している必要があります。

サンプル

```
<host>it3</host>
```

関連要素

親要素

- <web-deploy-path> 要素

子要素

- なし

<jndi-name> 要素

```
<element name="jndi-name" type="xsd:string" minOccurs="0"/>
```

この要素は、Web アプリケーションによって参照されるリソースの、JNDI サービス ルックアップ名を提供します。

サンプル

```
<web-app>
   <ejb-local-ref>
      <ejb-ref-name>ejb/OrderFulfillmentFacade</ejb-ref-name>
      <jndi-name>ejb/local/supplier/supplier/OrderFulfillmentFacade</jndi-name>
   </ejb-local-ref>
   <resource-ref>
      <res-ref-name>jms/TopicConnectionFactory</res-ref-name>
      <jndi-name>jms/xatcf</jndi-name>
   </resource-ref>
   <resource-env-ref>
      <resource-env-ref-name>jms/opc/InvoiceTopic</resource-env-ref-name>
      <jndi-name>jms/opc/InvoiceTopic</jndi-name>
   </resource-env-ref>
   <web-deploy-path>
      <service>HTTP</service>
      <engine>HTTP</engine>
      <host>*</host>
   </web-deploy-path>
   <web-deploy-path>
      <service>IIOP</service>
      <engine>IIOP</engine>
      <host>*</host>
   </web-deploy-path>
   <security-role>
      <role-name>administrator</role-name>
   </security-role>
</web-app>
```

関連要素

親要素

- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <resource-ref> 要素
- <resource-env-ref> 要素
- <message-destination-ref> 要素
- <message-destination> 要素

子要素

- なし

<message-destination> 要素

```
<element name="message-destination" type="borl:message-destinationType"
minOccurs="0" maxOccurs="unbounded"/>

<complexType name="message-destinationType">
  <sequence>
    <element name="message-destination-name" type="xsd:string"/>
    <element name="jndi-name" type="xsd:string"/>
  </sequence>
</complexType>
```

この要素は、メッセージの送信先（JMS のキューやトピックなど）を定義するために使用されます。この要素は、Web クライアント内に 1 つ以上ある、`message-destination-ref` 要素の `message-destination-link` と対応していなければなりません。各メッセージの送信先には、`message-destination-link` 値に一致する `message-destination-name` と、関連付けられている `jndi-name` が含まれます。

サンプル

```
<web-app>
  ...
  <message-destination>
    <message-destination-name>myAppQueue</message-destination-name>
    <jndi-name>jms/queues/TibcoQueue1</jndi-name>
  </message-destination>
  ...
</web-app>
```

関連要素

親要素

- <web-app> 要素

子要素

- <message-destination-name> 要素
- <jndi-name> 要素

<message-destination-name> 要素

```
<element name="message-destination-name" type="xsd:string"/>
```

この要素は、JMS キューやトピックなど、対象となるメッセージの送信先に割り当てられる論理名を指定します。この名前は、共通のターゲットである送信先を識別し、`message-destination-ref` 要素の `message-destination-link` とともに使用して、Web アプリケーションにおけるメッセージ フローを表します。

サンプル

標準 Web アプリケーション ディスクリプタ web.xml :

```
<web-app>
 ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
     <message-destination-type>javax.jms.Queue</message-destination-type>
     <message-destination-usage>Consumes</message-destination-usage>
     <message-destination-link>myAppQueue</message-destination-link>
   </message-destination-ref>
 ...
   <message-destination>
     <message-destination-name>myAppQueue</message-destination-name>
   </message-destination>
</web-app>
```

Borland Web アプリケーション ディスクリプタ web-borland.xml :

```
<web-app>
 ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
     <jndi-name>jms/queues/Queue1</message-destination-type>
   </message-destination-ref>
 ...
   <message-destination>
     <message-destination-name>myAppQueue</message-destination-name>
     <jndi-name>jms/queues/TibcoQueue</jndi-name>
   </message-destination>
</web-app>
```

`message-destination-link` を介して、このアプリケーションが、名前が **jms/StockQueue** の `message-destination-ref` に対して JNDI ルックアップを実行した場合、`message-destination` の `jndi-name` **jms/queues/TibcoQueue** が使用され、`jndi-name` **jms/queues/Queue1** は使用されてない点に注意してください。

関連要素

親要素

- <message-destination> 要素

子要素

- なし

<message-destination-ref> 要素

```
<element name="message-destination-ref" type="borl:message-destination-
refType" minOccurs="0" maxOccurs="unbounded"/>

<complexType name="message-destination-refType">
  <sequence>
    <element name="message-destination-ref-name" type="xsd:string"/>
    <element name="jndi-name" type="xsd:string"/>
  </sequence>
</complexType>
```

この要素は、JMS キューやトピックなど、メッセージの送信先リファレンスを定義するために使用されます。各 EJB リファレンスには、アプリケーションによって使用される `message-destination-ref-name` と、それに関連付けられた `jndi-name` が含まれています。

サンプル

```
<web-app>
   ...
   <message-destination-ref>
     <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
     <jndi-name>jms/queues/Queue1</message-destination-type>
   </message-destination-ref>
   ...
</web-app>
```

関連要素

親要素

- <web-app> 要素

子要素

- <message-destination-ref-name> 要素
- <jndi-name> 要素

<message-destination-ref-name> 要素

```
<element name="message-destination-ref-name" type="xsd:string"/>
```

この要素は、JMS キューやトピックなど、メッセージの送信先リファレンスにアクセスするために、Web アプリケーションが使用する論理名を指定します。この名前は、アプリケーション コンポーネントの java:comp/env への JNDI 相対名です。

サンプル

```
<web-app>
  ...
  <message-destination-ref>
    <message-destination-ref-name>jms/StockQueue</message-destination-
ref-name>
    <jndi-name>jms/queues/Queue1</message-destination-type>
  </message-destination-ref>
  ...
</web-app>
```

関連要素

親要素

- <message-destination-ref> 要素

子要素

- なし

<property> 要素

```
<element name="property" minOccurs="0" maxOccurs="unbounded">
  <complexType>
    <sequence>
      <element name="prop-name" type="xsd:string"/>
      <element name="prop-type" type="xsd:string"/>
      <element name="prop-value" type="xsd:string"/>
    </sequence>
  </complexType>
</element>
```

この要素は、Web アプリケーションに含まれるか、これによって参照されるさまざまなリソースのプロパティ値を指定するために使用されます。各 `property` エントリは、対応するサブ要素を使用して、プロパティの名前、型、および値を指定します。

サンプル

```
<property>
  <prop-name>vbroker.security.disable</prop-name>
  <prop-type>security</prop-type>
  <prop-value>false</prop-value>
</property>
```

関連要素

親要素

- <web-app> 要素

子要素

- <prop-name> 要素
- <prop-type> 要素
- <prop-value> 要素

<prop-name> 要素

```
<element name="prop-name" type="xsd:string"/>
```

設定するプロパティの名前を指定します。

サンプル

```
<prop-name>vbroker.security.disable</prop-name>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-type> 要素

```
<element name="prop-type" type="xsd:string"/>
```

設定するプロパティの型を指定します。

サンプル

```
<prop-type>security</prop-type>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-value> 要素

```
<element name="prop-value" type="xsd:string"/>
```

設定するプロパティの値を指定します。

サンプル

```
<prop-value>false</prop-value>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<resource-env-ref-name> 要素

```
<element name="resource-env-ref-name" type="xsd:string"/>
```

この要素は、Web アプリケーションがリソース環境リファレンスにアクセスするために使用する名前を提供します。

サンプル

```
<web-app>
   ...
   <resource-env-ref>
      <resource-env-ref-name>jms/StockQueue</resource-env-ref-name>
      <jndi-name>jms/Tibco/Queue1</jndi-name>
   </resource-ref>
   ...
</web-app>
```

関連要素

親要素

- <resource-env-ref> 要素

子要素

- なし

<resource-env-ref> 要素

```
<element name="resource-env-ref" type="borl:resource-env-refType"
minOccurs="0" maxOccurs="unbounded"/>

<complexType name="resource-env-refType">
 <sequence>
  <element name="resource-env-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string"/>
 </sequence>
</complexType>
```

この要素は、Web アプリケーションで使用されるリソース環境リファレンスを JNDI の名前にマップするために使用されます。各リソース環境リファレンスには、Bean によって使用される `res-env-ref-name` と、それに関連付けられた `jndi-name` が含まれています。

サンプル

```
<web-app>
   ...
   <resource-env-ref>
      <resource-env-ref-name>jms/StockQueue</resource-env-ref-name>
      <jndi-name>jms/Tibco/Queue1</jndi-name>
   </resource-ref>
   ...
</web-app>
```

関連要素

親要素

- <web-app> 要素

子要素

- <resource-env-ref> 要素
- <jndi-name> 要素

<resource-ref> 要素

```
<element name="resource-ref" type="borl:resource-refType" minOccurs="0"
maxOccurs="unbounded"/>

<complexType name="resource-refType">
 <sequence>
  <element name="res-ref-name" type="xsd:string"/>
  <element name="jndi-name" type="xsd:string"/>
 </sequence>
</complexType>
```

この要素は、Web アプリケーションによって使用されるリソース リファレンスを定義するために使用されます。各リソース リファレンスには、アプリケーションによって使用される `res-ref-name` と、それに関連付けられた `jndi-name` が含まれています。

サンプル

```
<web-app>
   <ejb-local-ref>
      <ejb-ref-name>ejb/OrderFulfillmentFacade</ejb-ref-name>
      <jndi-name>ejb/local/supplier/supplier/OrderFulfillmentFacade
            </jndi-name>
   </ejb-local-ref>
   <resource-ref>
      <res-ref-name>jdbc/CheckingDataSource</res-ref-name>
      <jndi-name>datasources/OracleDataSource</jndi-name>
   </resource-ref>
   <web-deploy-path>
      <service>HTTP</service>
      <engine>HTTP</engine>
      <host>*</host>
   </web-deploy-path>
   <web-deploy-path>
      <service>IIOP</service>
      <engine>IIOP</engine>
      <host>*</host>
   </web-deploy-path>
   <security-role>
      <role-name>administrator</role-name>
   </security-role>
</web-app>
```

関連要素

親要素

- <web-app> 要素

子要素

- <res-ref-name> 要素
- <jndi-name> 要素

<res-ref-name> 要素

```
<element name="res-ref-name" type="xsd:string"/>
```

この要素は、Web アプリケーションがリソース リファレンスにアクセスするために使用する名前を提供します。

サンプル

```
<web-app>
   <ejb-local-ref>
      <ejb-ref-name>ejb/OrderFulfillmentFacade</ejb-ref-name>
      <jndi-name>ejb/local/supplier/supplier/OrderFulfillmentFacade
            </jndi-name>
   </ejb-local-ref>
   <resource-ref>
      <res-ref-name>jdbc/CheckingDataSource</res-ref-name>
      <jndi-name>datasources/OracleDataSource</jndi-name>
   </resource-ref>
   <web-deploy-path>
      <service>HTTP</service>
      <engine>HTTP</engine>
      <host>*</host>
   </web-deploy-path>
   <web-deploy-path>
      <service>IIOP</service>
      <engine>IIOP</engine>
      <host>*</host>
   </web-deploy-path>
   <security-role>
      <role-name>administrator</role-name>
   </security-role>
</web-app>
```

関連要素

親要素

- <resource-ref> 要素

子要素

- なし

<role-name> 要素

```
<element name="role-name" type="xsd:string"/>
```

Web アプリケーションで使用される security-role のロール名。これは、BAS のデプロイ環境のロールにマップされます。

サンプル

```
<role-name>administrator</role-name>
```

関連要素

親要素

- <security-role> 要素

子要素

- なし

<security-role> 要素

```
<element name="security-role" type="borl:security-roleType" minOccurs="0"
maxOccurs="unbounded"/>

<complexType name="security-roleType">
 <sequence>
  <element name="role-name" type="xsd:string"/>
  <element name="deployment-role" type="xsd:string" minOccurs="0"/>
 </sequence>
</complexType>
```

Web アプリケーションのロール（`web.xml` にある）を Borland AppServer の deployment-role にマップします。

サンプル

```
<security-role>
      <role-name>administrator</role-name>
      <deployment-role>administrator</deployment-role>
</security-role>
```

関連要素

親要素

- <web-app> 要素

子要素

- <role-name> 要素
- <deployment-role> 要素

<service> 要素

```
<element name="service" type="xsd:string"/>
```

サービス名を設定します。これは、Tomcat の `server.xml` ファイルで定義されたサービスに対応している必要があります。

サンプル

```
<service>tomcatX</service>
```

関連要素

親要素

- <web-deploy-path> 要素

子要素

- なし

<web-app> 要素

```
<element name="web-app">
  <complexType>
    <sequence>
      <element name="context-root" type="xsd:string" minOccurs="0"/>
      <element name="ejb-ref" type="borl:ejb-refType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="ejb-local-ref" type="borl:ejb-local-refType" minOccurs="0"
               maxOccurs="unbounded"/>
      <element name="resource-ref" type="borl:resource-refType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="resource-env-ref" type="borl:resource-env-refType" minOccurs="0"
               maxOccurs="unbounded"/>
      <element name="message-destination-ref" type="borl:message-destination-refType" minOccurs="0"
               maxOccurs="unbounded"/>
      <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="web-deploy-path" type="borl:web-deploy-pathType" minOccurs="0"
               maxOccurs="unbounded"/>
      <element name="authorization-domain" type="xsd:string" minOccurs="0"/>
      <element name="security-role" type="borl:security-roleType" minOccurs="0"
               maxOccurs="unbounded"/>
      <element name="message-destination" type="borl:message-destinationType" minOccurs="0"
               maxOccurs="unbounded"/>
    </sequence>
  </complexType>
</element>
```

Web アプリケーションのデプロイメント ディスクリプタのルート ノード。このディスクリプタは、`web.xml` 標準デプロイメント ディスクリプタを拡張して、追加のプロパティを提供したり、Web アプリケーションがホストされる場所を正確に指定したり、アプリケーションのセキュリティ情報を提供できるようにします。

サンプル

```
<web-app>
   <authorization-domain>default</authorization-domain>
</web-app>
```

関連要素

親要素

- なし

子要素

- <context-root> 要素
- <resource-env-ref> 要素
- <resource-ref> 要素
- <ejb-ref> 要素
- <ejb-local-ref> 要素
- <property> 要素
- <web-deploy-path> 要素
- <authorization-domain> 要素
- <security-role> 要素
- <message-destination-ref> 要素
- <message-destination> 要素

<web-deploy-path> 要素

```
<element name="web-deploy-path" type="borl:web-deploy-pathType"
minOccurs="0" maxOccurs="unbounded"/>

<complexType name="web-deploy-pathType">
  <sequence>
    <element name="service" type="xsd:string"/>
    <element name="engine" type="xsd:string"/>
    <element name="host" type="xsd:string"/>
  </sequence>
</complexType>
```

Tomcat の `server.xml` ファイルでは、特定のサービスの下にある 1 つ以上のエンジンの下に 1 つ以上のホストを定義できます。Tomcat コンテナの下の Web アプリケーションをデプロイする場所を正確に指定する場合は、この要素を使用します。

サンプル

```
<web-deploy-path>
   <service>tomcatX</service>
   <engine>cyrpi</engine>
   <host>it3</host>
</web-deploy-path>
```

関連要素

親要素

- <web-app> 要素

子要素

- <service> 要素
- <engine> 要素
- <host> 要素

第 7 章

DAR モジュール : jndi-definitions.xml

XSD: jndi-definitions.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://support.borland.com/appserver/xml/ns/j2ee" xmlns:xsd="http://www.w3.org/2001/
XMLSchema" xmlns:borl="http://support.borland.com/appserver/xml/ns/j2ee" xmlns="http://www.w3.org/2001/
XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.4">
 <element name="jndi-definitions">
  <complexType>
   <sequence>
    <element name="visitransact-datasource" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element ref="borl:jndi-name"/>
       <element name="driver-datasource-jndiname" type="xsd:string"/>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
    <element name="driver-datasource" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element ref="borl:jndi-name"/>
       <element name="datasource-class-name" type="xsd:string"/>
       <element name="log-writer" type="xsd:string" minOccurs="0"/>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
    <element name="jndi-object" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <sequence>
       <element ref="borl:jndi-name"/>
       <element name="class-name" type="xsd:string"/>
       <element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
     </complexType>
    </element>
   </sequence>
```

```
 </complexType>
</element>
<element name="jndi-name" type="xsd:string"/>
<complexType name="propertyType">
 <sequence>
  <element name="prop-name" type="xsd:string"/>
  <element name="prop-type" type="xsd:string"/>
  <element name="prop-value" type="xsd:string"/>
 </sequence>
</complexType>
</schema>
```

<class-name> 要素

```
<xsd:element name="class-name" type="xsd:string"/>
```

JMS サービス プロバイダから提供され、ライブラリとして BAS パーティションにデプロイされる接続ファクトリ クラスの名前。

サンプル

```
<class-name>progress.message.jclient.QueueConnectionFactory</class-name>
```

関連要素

親要素

- <jndi-object> 要素

子要素

- なし

<datasource-class-name> 要素

```
<xsd:element name="datasource-class-name" type="xsd:string"/>
```

リソース ベンダーから提供される接続ファクトリ クラスの名前を指定します。クラス自体は、ライブラリとして BAS パーティションにデプロイされます。

サンプル

```
<datasource-class-name>oracle.jdbc.pool.OracleConnectionPoolDataSource</
datasource-class-name>
```

関連要素

親要素

- <driver-datasource> 要素

子要素

- なし

<driver-datasource> 要素

```
<xsd:element name="driver-datasource" minOccurs="0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="borl:jndi-name"/>
      <xsd:element name="datasource-class-name" type="xsd:string"/>
      <xsd:element name="log-writer" type="xsd:string" minOccurs="0"/>
      <xsd:element name="property" type="borl:propertyType" minOccurs=
            "0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

この要素は、`<visitransact-datasource>` で開始されたデータソース定義の残り半分として、ドライバに関する情報を指定します。ここではドライバの JNDI 名を指定します。これは、`defined in <visitransact-datasource>` で定義されたデータソースの `<driver-datasource-jndiname>` と同じである必要があります。データソースドライバのクラス名、ログ動作（該当する場合）、および JDBC リソースに固有のプロパティ（ユーザー名、パスワードなど）も指定します。

サンプル

```
<driver-datasource>
      <jndi-name>serial://datasources/OracleDriver</jndi-name>
      <datasource-class-
name>oracle.jdbc.pool.OracleConnectionPoolDataSource</datasource-class-
name>
      <property>
         <prop-name>user</prop-name>
         <prop-type>String</prop-type>
         <prop-value>MisterKittles</prop-value>
      </property>
</driver-datasource>
```

関連要素

親要素

- <jndi-definitions> 要素

子要素

- <jndi-name> 要素
- <datasource-class-name> 要素
- <log-writer> 要素
- <property> 要素

<driver-datasource-jndiname> 要素

```
<xsd:element name="driver-datasource-jndiname" type="xsd:string"/>
```

データベース ベンダーによって提供されたドライバ クラスの JNDI 名。アプリケーションをホストとなる BAS パーティションには、このドライバ クラスを含む Java ライブラリがデプロイされる必要があります。この要素の値は、JDBC リソース定義の残りの部分を構成する `<driver-datasource>` 要素の子である `<jndi-name>` によって参照される名前と同じです。

サンプル

```
<driver-datasource-jndiname>serial://datasources/OracleDriver</driver-datasource-jndiname>
```

関連要素

親要素

- <visitransact-datasource> 要素

子要素

- なし

<jndi-definitions> 要素

```
<xsd:element name="jndi-definitions">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="visitransact-datasource" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element ref="borl:jndi-name"/>
      <xsd:element name="driver-datasource-jndiname" type="xsd:string"/>
      <xsd:element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="driver-datasource" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element ref="borl:jndi-name"/>
      <xsd:element name="datasource-class-name" type="xsd:string"/>
      <xsd:element name="log-writer" type="xsd:string" minOccurs="0"/>
      <xsd:element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="jndi-object" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element ref="borl:jndi-name"/>
      <xsd:element name="class-name" type="xsd:string"/>
      <xsd:element name="property" type="borl:propertyType" minOccurs="0" maxOccurs="unbounded"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

J2EE リソース接続ファクトリ オブジェクトを JNDI 定義モジュールの一部としてデプロイすると、J2EE リソース接続ファクトリ オブジェクトが JNDI に接続されます。この定義モジュールは、他の J2EE 標準 Java アーカイブ型に似ており、拡張子 .dar で終わります。そのため、このモジュールのことを DAR ともいいます。このモジュールは、JAR、WAR、RAR などの標準 J2EE モジュール タイプに追加されます。このモジュールはまた、EAR の一部としてパッケージする場合と、スタンドアロンでデプロイする場合があります。

提供する必要がある DAR の内容は、jndi-definitions.xml という名前の XML ディスクリプタ ファイルだけです。このファイルには、JNDI 名前空間に連結されるすべてのデータソース定義が含まれます。

`<jndi-definitions>` 要素は、スキーマのルート ノードです。`<visitransact-datasource>` および `<driver-datasource>` 子ノードを使って JDBC 接続ファクトリ オブジェクトを定義し、`<jndi-object>` 子ノードを使って JMS リソース接続ファクトリ オブジェクトを定義します。

サンプル

```
<jndi-definitions>
   <visitransact-datasource>
      <jndi-name>serial://datasources/Oracle</jndi-name
      <driver-datasource-jndiname>serial://datasources/OracleDriver</driver-
datasource-jndiname>
      <property>
         <prop-name>connectionType</prop-name>
         <prop-type>Enumerated</prop-type>
         <prop-value>Direct</prop-value>
      </property>
   </visitransact-datasource>
   <driver-datasource>
      <jndi-name>serial://datasources/OracleDriver</jndi-name>
      <datasource-class-
name>oracle.jdbc.pool.OracleConnectionPoolDataSource</datasource-class-
name>
      <property>
         <prop-name>user</prop-name>
         <prop-type>String</prop-type>
         <prop-value>MisterKittles</prop-value>
      </property>
   </driver-datasource>
</jndi-definitions>
```

関連要素

親要素

- なし

子要素

- <visitransact-datasource> 要素
- <driver-datasource> 要素
- <jndi-object> 要素

<jndi-name> 要素

```
<xsd:element name="jndi-name" type="xsd:string"/>

<xsd:element ref="borl:jndi-name"/>
```

JNDI が参照するデータソースの名前。エンタープライズ Bean のリソース リファレンスにもある名前です。

サンプル

```
<jndi-name>serial://datasources/Oracle</jndi-name>
```

関連要素

Parents

- <visitransact-datasource> 要素
- <driver-datasource> 要素
- <jndi-object> 要素
- <jndi-definitions> 要素

子要素

- なし

<jndi-object> 要素

```
<xsd:element name="jndi-object" minOccurs="0" maxOccurs="unbounded">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element ref="borl:jndi-name"/>
   <xsd:element name="class-name" type="xsd:string"/>
   <xsd:element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

You define the `<jndi-object>` 要素を定義して、JNDI に JMS 接続ファクトリを登録します。その子ノードを使用して、JMS 接続を確立するための JNDI 検索、接続ファクトリクラス、およびそのクラスに渡す必要がある JMS プロバイダに固有のプロパティを提供します。

サンプル

```
<jndi-object>
   <jndi-name>serial://jms/message</jndi-name>
   <class-name>progress.message.jclient.QueueConnectionFactory</class-
name>
   <property>
      <prop-name>connectionURLS</prop-name>
      <prop-type>String</prop-type>
      <prop-value>localhost:2506</prop-value>
   </property>
   <property>
      <prop-name>sequential</prop-name>
      <prop-type>Boolean</prop-type>
      <prop-value>false</prop-value>
   </property>
   <property>
      <prop-name>loadBalancing</prop-name>
      <prop-type>Boolean</prop-type>
      <prop-value>true</prop-value>
   </property>
</jndi-object>
```

関連要素

親要素

- <jndi-definitions> 要素

子要素

- <jndi-name> 要素
- <class-name> 要素
- <property> 要素

<log-writer> 要素

```
<xsd:element name="log-writer" type="xsd:string" minOccurs="0"/>
```

この要素を使用して、一部のベンダー接続ファクトリ クラスの詳細モードを有効にできます。このプロパティの使用方法については、リソースのマニュアルを参照してください。

サンプル

```
<log-writer>True</log-writer>
```

関連要素

親要素

- <driver-datasource> 要素

子要素

- なし

<property> 要素

```
<xsd:element name="property" type="borl:propertyType" minOccurs="0"
maxOccurs="unbounded"/>

<xsd:complexType name="propertyType">
   <xsd:sequence>
     <xsd:element name="prop-name" type="xsd:string"/>
     <xsd:element name="prop-type" type="xsd:string"/>
     <xsd:element name="prop-value" type="xsd:string"/>
   </xsd:sequence>
 </xsd:complexType>
```

この要素は、アーカイブまたはそのコンポーネントに含まれるか、それらから参照されるさまざまなリソースのプロパティ値を指定するために使用されます。各 `property` エントリは、対応するサブ要素を使用して、プロパティの名前、型、および値を指定します。

サンプル

```
<property>
   <prop-name>vbroker.security.disable</prop-name>
   <prop-type>security</prop-type>
   <prop-value>false</prop-value>
</property>
```

関連要素

親要素

- <visitransact-datasource> 要素
- <driver-datasource> 要素
- <jndi-object> 要素
- <jndi-definitions> 要素

子要素

- <prop-name> 要素
- <prop-type> 要素
- <prop-value> 要素

<prop-name> 要素

```
<xsd:element name="prop-name" type="xsd:string"/>
```

設定するプロパティの名前を指定します。

サンプル

```
<prop-name>vbroker.security.disable</prop-name>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-type> 要素

```
<xsd:element name="prop-type" type="xsd:string"/>
```

設定するプロパティの型を指定します。

サンプル

```
<prop-type>security</prop-type>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<prop-value> 要素

```
<xsd:element name="prop-value" type="xsd:string"/>
```

設定するプロパティの値を指定します。

サンプル

```
<prop-value>false</prop-value>
```

関連要素

親要素

- <property> 要素

子要素

- なし

<visitransact-datasource> 要素

```
<xsd:element name="visitransact-datasource" minOccurs="0"
maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="borl:jndi-name"/>
      <xsd:element name="driver-datasource-jndiname" type="xsd:string"/>
      <xsd:element name="property" type="borl:propertyType" minOccurs=
            "0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

アプリケーション コードが検索するデータソースを定義します。データソースの JNDI 名、そのドライバ、およびそのドライバに渡す必要があるプロパティを提供します。これを定義したら、`<driver-datasource>` 兄弟要素で、そのドライバに関する情報を提供する必要があります。

サンプル

```
<visitransact-datasource>
  <jndi-name>serial://datasources/Oracle</jndi-name
 <driver-datasource-jndiname>serial://datasources/OracleDriver</driver-
datasource-jndiname>
  <property>
   <prop-name>connectionType</prop-name>
   <prop-type>Enumerated</prop-type>
   <prop-value>Direct</prop-value>
  </property>
</visitransact-datasource>
```

関連要素

親要素

- <jndi-definitions> 要素

子要素

- <jndi-name> 要素
- <driver-datasource> 要素
- <property> 要素

索引

記号

A

B

C

D

E

J

R

W

か

き

こ

さ

そ

て

と