Oracle® Retail

Functional Artifacts Guide

Release 16.0

E78846-01

December 2016

ORACLE®

Oracle Retail Functional Artifacts Guide, Release 16.0

E78846-01

Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

5 XML Standards and Conventions

6 Customizing and Extending Business Objects

7 Localizing Retail Business Objects

8 Importing Retail Business Objects

A   References

Glossary

Send Us Your Comments

Oracle Retail Functional Artifacts Guide, Release 16.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?

- Did you understand the context of the procedures?

- Did you find any errors in the information?

- Does the structure of the information help you with your tasks?

- Do you need different information or graphics? If so, where, and in what format?

- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Applications Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

x

Preface

The *Oracle Retail Artifacts Guide* provides detailed information about how Oracle Retail artifacts are designed and packaged for release, as well as the standards used in their creation.

Audience

This *Oracle Retail Artifacts Guide* is intended for users of Oracle Retail applications, particularly application integrators, implementation staff, and retail IT personnel.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Integration Bus documentation set:

- *Oracle Retail Integration Bus Implementation Guide*

- *Oracle Retail Integration Bus Installation Guide*

- *Oracle Retail Integration Bus Operations Guide*

- *Oracle Retail Integration Bus Release Notes*

- *Oracle Retail Integration Bus Hospital Administration Guide*

- *Oracle Retail Integration Bus Security Guide*

- *Oracle Retail Integration Bus Support Tools Guide*

- *Oracle Retail Enterprise Integration Guide*

- *Oracle Retail Integration Bus Integration Gateway Services Guide*

- *Oracle Retail Functional Artifact Generator Guide*
- *Oracle Retail Service-Oriented Architecture Enabler Tool Guide*
- *Oracle Retail Integration Bus Java Messaging Service (JMS) Console Guide*
- *Oracle Retail Integration Bus Data Model*
- *Oracle Retail Payload Mapper Guide*

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 16.0) or a later patch release (for example, 16.0.1). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction

The purpose of this document is to provide detailed information about the following:

- How artifacts are designed and packaged for release, as well as the standards used for their creation.

- Version strategy and implementation of that strategy, as well as the standards used for base Business Objects.

- Customization strategy and implementation of that strategy, including extension of base Business Objects.

Overview

The Oracle Retail Business Object (BO) is the name given to the logical representation of an Oracle Retail Business Entity. The definition of that object is in the form of an XML Schema (XSD). The schema represents the common object definition for business concepts, such as account, supplier, purchase order, and item.

Within Oracle Retail, the messages that flow between the retail applications are defined statically through these XML schemas (XSDs). These Business Objects are the message payloads, or the basis of message payloads for the Oracle Retail integration products, such as the Oracle Retail Integration Bus (RIB) and RIB Integration Gateway Services (IGS) and Web Service Providers created by the Retail SOA Enabler tool.

The Oracle Retail integration infrastructure products work with multiple technologies (for example, Java EE and PL/SQL). So they have different ways of representing the same functional XML message structure in the various technologies. The Business Object XSD is the common source used to generate the various manifestations of the BO (payloads) used within the infrastructure products and to ultimately communicate the Business Objects appropriately to the Oracle Retail API technology.

The XSDs are strict, and they are used by Oracle Retail standard tools to produce the design time physical objects used by the application's API technology (PL/SQL or Java), as well as the runtime validations used by the various integration components.

> **Note:** For more information, see the *Oracle Retail Functional Artifact Generator Guide*.

Retail Functional Artifacts is the collective name given to the package containing the base Business Object XSDs and the various representations of the same structure/definition in these different technologies. These representations include the following:

- For Java EE, the artifacts are JAXB Java Beans. JAXB is a standard Java XML binding technology. It provides the mechanism to convert XML instances to Java objects (and vice versa) in a standard way. The JavaEE Web service infrastructure internally uses JAXB to marshall and unmarshall the SOAP messages. For every Business Object XSD, there is a corresponding JAXB bean.

- For PL/SQL retail applications, the artifacts are Oracle Objects. These objects are user defined database objects that define the XML message structure inside the database. For every Business Object XSD, there is a corresponding Oracle Object SQL defined.

2

Packaging Structure and Standards

This chapter describes the physical packaging structure of the Retail Functional Artifacts. There are two Functional Artifacts packages:

- RibFuncArtifact16.0.0ForAll16.0.0Apps_eng_ga.tar

- RetailFuncArtifact16.0.0ForAll16.0.0Apps_eng_ga.tar

The RibFuncArtifact contains additional artifacts specific to the RIB application, such as TAFR related artifacts. The RetailFuncArtifact contains only product independent artifacts. All content is versioned and follows the standards as described in Chapter 4, "Versioning Artifacts."

General Packaging Structure

The Retail Functional Artifacts are physically packaged as a directory structure. The following diagram illustrates the structure. Standard UNIX directory formatting is used within the rules associated with the physical packaging of the directory.

```
/retail/integration/base/
                    bo/  (Business Objects)
                        BO  (Object Name)
                           v[1]  (Version - latest)
                             BO.xsd   (BO Schema)
                        LocOfBO  (Object Name)
                           v[1]  (Version - latest)
                             LocOfBO.xsd   (Localization hook to BO Schema)

                /custom/
                    bo/  (Business Objects)
                        ExtOf<BO>  (Object Name)
                           v[1]  (Version - latest)
                             ExtOf<BO>.xsd   (Custom BO Schema)
                        EOfBr<BO>  (Object Name)
                           v[1]  (Version - latest)
                             EOfBr<BO>.xsd   (Custom Localization BO Schema)
                        EOfIn<BO>  (Object Name)
                           v[1]  (Version - latest)
                             EOfIn<BO>.xsd   (Custom Localization BO Schema)

                /localization/
                    bo/  (Business Objects)
                        Br<BO>  (Object Name)
                           v[1]  (Version - latest)
                                      Br<BO>.xsd  (Localization BO
Schema)
                        In<BO>  (Object Name)
```

```
v[1]  (Version - latest)
  In<BO>.xsd   (Localization BO Schema)
```

> **Note:** For a GA deliverable, the custom and localization directories
> are created and populated by default with customized and localized
> XSD for twelve business objects namely, FulfilOrdCustDesc,
> FulfilOrdPmtDesc, PayTermDesc, PayTermRef, ReportLocDesc,
> ReportLocRef, SupplierColDesc, SupplierColRef, SupplierDesc,
> SupplierRef, XItemDesc, and XStoreDesc. For remaining retail
> business objects, a new tool that is part of the Functional Artifact
> Generator called the Template Creator is provided to create the
> placeholder XSDs and the import hooks in the base objects.
>
> See the *Oracle Retail Functional Artifact Generator Guide* for more
> information.

3

Business Object Structure

This chapter describes the directories that contain the Oracle Retail Business Objects.

Base and Custom Directories

The base directory contains the Oracle Retail Business Objects, as delivered with each release and used by Oracle Retail applications within the integration products. The packaging structure reflects the namespace and the versioning within the namespace.

For example:

The ASNInBO.xsd namespace:

```
targetNamespace="http://www.oracle.com/retail/integration/base/bo/ASNInBO/v1"
version="1.0"
xmlns=http://www.oracle.com/retail/integration/base/bo/ASNInBO/v1
```

The ASNInBO.xsd within the packaging structure:

```
retail/integration/base/bo/ASNInBO/v1/ASNInBO.xsd
```

Custom and Localization Directories and Namespaces

Additional directories, placeholder XSDs and hooks in the base XSDs are optionally created using the Template Creator tool.

The Template Creator creates the custom and localization directories and hooks for a specific business object (for example, ASNIn).

> **Note:** For more information, see the *Oracle Retail Artifact Generator Guide.*

On demand for the business object specified, the Template Generator will create the hooks to customize or localize the Oracle Retail Business Object.

For example:

The LocOfASNInBO.xsd namespace:

```
targetNamespace="http://www.oracle.com/retail/integration/base/bo/LocOfASNInBO/v1"
version="1.0"
xmlns=http://www.oracle.com/retail/integration/base/bo/LocOfASNInBO/v1
```

The LocOfASNInBO.xsd within the packaging structure:

```
retail/integration/base/bo/LocOfASNInBO/v1/LocOfASNInBO.xsd
```

The custom business objects will be contained within the Custom sub directory, mimicking the base directory but prefaced ExtOf, EOf[locale], where locale is the ISO standard alpha-2 country code.

For example:

- The ExtOfASNInBO.xsd namespace:

```
targetNamespace="http://www.oracle.com/retail/integration/custom/bo/ExtOfASNInB
O/v1"
    version="1.0"
    xmlns=http://www.oracle.com/retail/integration/custom/bo/ExtOfASNInBO/v1
```

    The ExtOfASNInBO.xsd within the packaging structure:

```
retail/integration/custom/bo/ExtOfASNInBO/v1/ExtOfASNInBO.xsd
```

- The EOfBrASNInBO.xsd namespace

```
targetNamespace="http://www.oracle.com/retail/integration/custom/bo/EOfBrASNInB
O/v1"
    version="1.0"
    xmlns=http://www.oracle.com/retail/integration/custom/bo/EOfBrASNInBO/v1
```

The EOfBrASNInBO.xsd within the packaging structure.

```
retail/integration/custom/bo/EOfBrASNInBO/v1/EOfBrASNInBO.xsd
```

The Localization directories will be contained within the Localization subdirectory, mimicking the base directory, but prefaced within the ISO standard alpha-2 country code.

For example:

The BrASNInBO.xsd namespace:

```
targetNamespace="http://www.oracle.com/retail/integration/localization/bo/BrASNInB
O/v1"
    version="1.0"
    xmlns=http://www.oracle.com/retail/integration/localization/bo/BrASNInBO/v1
```

The BrASNInBO.xsd within the packaging structure.

```
retail/integration/localization/bo/BrASNInBO/v1/BrASNInBO.xsd
```

Within one artifact deliverable (packaged release) there will be only the latest version of each BO XSD file. For a breaking change, the XSD is moved to a directory with the same name as the version number in the changed namespace.

4

Versioning Artifacts

This chapter describes versioning strategies and standards.

Common Versioning Strategies

The following describes types of versioning strategies, as explained in Thomas Erl's book, *Web Service Contract Design and Versioning for SOA*.

Strict Versioning

For strict versioning, any compatible or incompatible change results in a new version of the service contract. This type of versioning does not support backward or forward compatibility (for example, new change and new contract).

Flexible Versioning

For flexible versioning, an incompatible change results in a new version of the service contract, and the contract supports backward compatibility but not forward compatibility.

Loose Versioning

For loose versioning, any incompatible change results in a new version of the service contract, and the contract supports backward and forward compatibility.

Comparison Chart

The following chart compares the types of versioning strategies listed above. The values in the Characteristics column are described as follows:

- Strictness: The rigidity of the contract versioning strategy.

- Governance Impact: The amount of governance burden imposed by the strategy.

- Complexity: The overall complexity of the versioning process.

Characteristic	Strict Strategy	Flexible Strategy	Loose Strategy
Strictness	High	Medium	Low
Governance Impact	High	Medium	High
Complexity	Low	Medium	High

Oracle Retail Versioning Strategy

The version approach proposed for Oracle Retail is the Flexible Strategy, where an incompatible change results in a new version of the service contract, and the contract supports backward compatibility but not forward compatibility.

Reasons for Versioning an Artifact

For this proposal, there are two primary reasons to version an artifact:

- To recognize that an artifact as changed.

- To recognize whether the change is *breaking* or *non-breaking*, as described below.

Breaking Changes

A breaking change is also known as non-backward compatibility. For example, newer versions of producers based on the same Business Object will break existing consumers and require a regeneration of the client from the new wsdl.

Non-backward compatible changes include the following:

- Removing an operation.

- Renaming an operation.

- Changing the parameters of an operation, such as data type or order.

Non-breaking Changes

A non-breaking change is also known as backward compatibility. For example, newer versions of producers based on the same Business Object can be deployed without breaking existing consumers.

Backward compatible changes include the following:

- New operations.

- New data types.

- New optional fields to existing data types.

- Type expansion.

Versioning Retail Business Objects Schemas (XSDs)

This section provides examples of versioning Retail Business Objects (BO) schemas (XSD)s.

Versioning Approach for XSDs

The XSD is versioned by an attribute (version) and the namespace. The namespace is the key versioning point.

XSD Version Attribute

Topic	Description
Recommendation	Each XSD has a version attribute.

Topic	Description
Rationale	The version attribute is used to indicate a change has occurred to the XSD. It is a simple counter that starts at 1.00 and bumps when any change is applied.
Example	targetNamespace="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1" version="1.0" xmlns="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1"

XSD Version Attribute Usage

Topic	Description
Recommendation	For any change, breaking or non-breaking, the version attribute is incremented.
Rationale	For a non-breaking change, the version will increment the point. For a breaking change, the version will bump. The breaking version bump equates to the same version reflected in the namespace.
Example - Non-breaking Change	Original targetNamespace="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1" version="1.0" xmlns="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1" Non-breaking Change targetNamespace="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1" version="1.1" xmlns="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1" Breaking Change targetNamespace="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v2" version="2.0" xmlns="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v2"

XSD Namespace Versioning

Topic	Description
Recommendation	XSD namespace is versioned.
Rationale	For breaking changes, the BO XSD namepsace is version controlled.

Topic	Description
Example	The following example shows how the versioning appears, before and after.
	Before
	targetNamespace="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1"
	version="1.0"
	xmlns="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v1"
	After
	targetNamespace="http://www.oracle.com/retail/integration/base/bo/PayTermDesc/v2"
	version="2.0"
	xmlns="http://www.oracle.com/retail/integration/base/bo/PayTermBO/v2"

Parent/Child Versioning

Topic	Description
Recommendation	For Parent/Child versioning, if a child version changes (a breaking change) the parent version also must change.
Rationale	If a parent imports an XSD and it has a breaking change, the parent also will have a breaking change. So the versioning of the parent should reflect that.

Multiple Versioning

Topic	Description
Recommendation	The Retail Functional Artifacts directory structure does not support multiple versions of a considered XSD is any of its subdirectory.
Rationale	Oracle Retail PL/SQL applications support only single version of a business object.
Example	The following directory structure depicting multiple versions of a considered XSD is not allowed. Only one version directory can exist.
	/retail/integration/base/
	bo/ (Business Objects)
	BO (Object Name)
	v[1] (Version - latest)
	BO.xsd (BO Schema)
	v[2] (Version - latest)
	BO.xsd (BO Schema)

5

XML Standards and Conventions

This chapter defines standards in the areas of naming conventions, XML standards, and customization/extension.

The standards in this document are for the artifacts used in Oracle Retail integration products, such as the RIB, Integration Gateway Services and Web Service Providers created by the Retail SOA Enabler tool. For Oracle AIA this section applies to edge-application Web services and integration patterns. These standards do not overlap the AIA standards for middle layer integrations. For those integrations, the AIA standards are directly applicable.

The standards and conventions described in this section are retail domain specific. The naming conventions and standards are only patterned after AIA concepts.

XML Standards and Conventions

This section describes XML standards and conventions.

The underscore (_) must be used for readability in naming simple elements

This standard is a significant deviation from generally accepted WS standards, including AIA. The underscore is used by the Business Objects in the RIB and by Oracle Retail PL/SQL applications.

Topic	Description
Recommendation	The underscore must be used for readability in naming simple elements, with no spaces or hyphens between words; lower camel case cannot be used (for example, supplierName). Lower case must be used. The simple element names must not be more than 30 characters in length.
Rationale	This is not the generally accepted recommendation in other XML standards.
	Because Business Objects are translated into Oracle Objects for use within the database by PL/SQL APIs, all elements are made uppercase by default. Readability is enhanced by using the underscore instead of the industry standard lower camel case.
Example	`<xsd:element name="supplier_name" type="varchar2240"/>`

Names must not contain special characters

Topic	Description
Recommendation	Names must not contain special characters.

Topic	Description
Rationale	This is recommended to be similar to other XML standards. Note that the underscore is allowed. This is not the same as other standards.
Examples	`space, '-', '.', '$', '%', '#',<, >, &, ?`

UpperCamel case must be used for naming complex types

Topic	Description
Recommendation	Upper camel case must be used for naming complex types with no spaces, hyphens or underscores between words. The complex type elements must not be more than 22 characters in length.
Rationale	This is recommended to be similar to other XML standards..
Example	`<xsd:complexType name="SupplierType"/>`

Avoid XML schema keywords as element names

Topic	Description
Recommendation	Avoid XML schema keywords as element names. Do not use XSD keywords, such as element, sequence, and complexType as element names.
Rationale	This is recommended to be similar to other XML standards.
Example	`<xs:element name="element" type="xs:integer"/>`

Avoid Java keywords

Topic	Description
Recommendation	Avoid Java keywords as element names.
Rationale	This is recommended to be similar to other XML standards.
Examples	abstract, assert, boolean, break, byte, case, catch, char, class, const, continue, default, do, double, else, enum, extends, false, final, finally, float, for, goto, if, implements, import, instanceof, int, interface, long, native, new, null, package, private, protected, public, return, short, static, strictfp, super, switch, synchronized, this, throw, throws, transient, true, try, void, volatile, while.

Avoid PLSQL keywords

Topic	Description
Recommendation	Avoid PLSQL keywords as element names.
Rationale	This is recommended to be similar to other XML standards.
Example	`comment,clob`

Avoid having numeric characters in the name

Topic	Description
Recommendation	Avoid having numeric characters in the name.

Topic	Description
Rationale	This is recommended to be similar to other XML standards.
Example	`<xsd:element name="supplier1Name" type="varchar2240"/>`

All XSD elements should be annotated

Topic	Description
Recommendation	The definitions of all elements/attributes/enumerations should be included in the schema as notations. Each XSD file should declare as a namespace: `xmlns:retailDoc="http://www.w3.org/2001/XMLSchema`
Rationale	This is recommended to be similar to other XML standards. The XSD should be the source of truth and include the business functional documentation of the element. The annotation standard should be independent of any repository specific requirements. The retailDoc namespace will allow intelligent processing in the XML/XSD parser in the future. There can be different behavior for a documentation element when the prefix is retailDoc versus anything else (including xs: or xsd:), even though all of them belong to same namespace.

Example	`xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"`
	`            <xs:element name="unit_qty" type="number12_4">`
	`                <retailDoc:annotation>`
	`                    <retailDoc:documentation>Contains the number of items`
	`expected to be received based on the supplier's ASN for this Item/Shipment`
	`combination.</retailDoc:documentation>`
	`                </retailDoc:annotation>`
	`            </xs:element>`

All XSDs should use the date type, xsd:dateTime, and the XML format, 2002-05-30T09:00:00

Topic	Description
Recommendation	All XSDs should use the date type, xsd:dateTime, and the XML format, 2002-05-30T09:00:00
Rationale	This is recommended to be similar to other XML standards.
Example	`<xs:element name="ship_date" type="xs:dateTime">`

All XSDs should use only these supported types

date: dateTime

boolean: boolean

text: varchar2N

integer: number6

decimal: numberN_N

long: number10

example:

varchar24

number12_4

Non top level complex type can not be empty

Topic	Description
Recommendation	Non top level CompleType cannot be empty, that is in xml we cannot have something like `<ExtOfReportLocDesc></ExtOfReportLocDesc>` or `<ExtOfReportLocDesc/>. So validation error is correct.`
Rationale	This is recommended to be similar to other XML standards.
Example	`<ExtOfReportLocDesc></ExtOfReportLocDesc>`

Retail Business Objects Schema

The logical representation of an Oracle Retail business entity is known as the Oracle Retail Business Object. For the purposes of this document, it describes the definition of that object in the form of an XML Schema (XSD). The schema represents the common object definition for business concepts such as Account, Supplier, Purchase Order, and Item.

Business Object (XSD) names should not contain names

Topic	Description
Recommendation	The Business Object name should not contain message related operations or actions, such as Request or Response.
Rationale	The name should be only a business noun and should not necessarily reference the usage of the object.
Example	Do not use Business Object names like ItemRequest.xsd or ItemResponse.xsd.

Business Object (XSD) names should not contain CRUD operations

Topic	Description
Recommendation	The Business Object name should not contain message related operations or actions, such as Create, Update, Process, and Get.
Rationale	The name should be only a business noun and should not necessarily reference the usage of the object CRUD operations.
Example	Do not use Business Object names like ItemCreate.xsd or ItemMod.xsd.

The Business Object (XSD) file must match the name of the top level element

Topic	Description
Recommendation	Every XSD file must define only one top level/global element, and the file name must match the name of the top level element.
Rationale	This is recommended to be similar to other XML standards.

Topic	Description

Example	`SupplierBO.xsd`
	   `<xs:element name="SupplierBO">`
	       `<retailDoc:annotation>`
	       `<retailDoc:documentation></retailDoc:documentation>`
	       `</retailDoc:annotation>`
	     `<xs:complexType>`

Each Business Object (XSD) must have its own namespace

Topic	Description
Recommendation	Each Business Object must have its own namespace.
Rationale	This is to eliminate conflicts when a service consumer will consume multiple from the same provider and the generate client side code should not have conflict. Each BO having its own namespace also allows versioning of each BO. The file will be located in a directory structure that matches the path portion of the namespace URLs for the top level type that is defined in that file.

Example	`targetNamespace="http://www.oracle.com/retail/integration/base/bo/SupplierBO/v1"`
	`version="1.0"`
	`xmlns="http://www.oracle.com/retail/integration/base/bo/SupplierBO/v1"`

Each Business Object (XSD) namespace should not contain application identifiers

Topic	Description
Recommendation	Each Business Object namespace should not contain application identifiers.
Rationale	Business Objects are not tied to any specific retail applications. These are generic business objects that belong to the full retail domain. So the Business Object namespace should not contain application identifiers.

Example	`targetNamespace="http://www.oracle.com/retail/integration/base/bo/SupplierBO/v1"`
	`version="1.0"`
	`xmlns="http://www.oracle.com/retail/integration/base/bo/SupplierBO/v1"`

Each Business Object (XSD) root should be annotated

Topic	Description
Recommendation	The XSD should be the source of truth and include the business functional documentation of the element.
Rationale	This is recommended to be similar to other XML standards. The annotation standard is independent of any repository specific requirements for the Business Objects.

Example	`<xs:element name="SupplierBO">`
	    `< retailDoc:annotation>`
	      `< retailDoc:documentation>`
	        `Merchandise Suppliers are suppliers of goods and services to be`
	`sold by the retailer to customers`
	      `</ retailDoc:documentation>`
	    `</ retailDoc:annotation>`

Avoid Business Object (XSD) to have maxOccurs="unbounded" for elements

Topic	Description
Recommendation	Avoid maxOccurs="unbounded" unless it is absolutely required.
Rationale	Setting a maxOccurs value to unbounded can lead to resources exhaustion and ultimately a denial of service.
	Processing XML documents can be expensive. Attackers can take advantage of the schemas that allow unbounded elements by supplying an application with a very large number elements causing the application to exhaust system resources.
Example	The following is an example of a schema that allows unbounded bar elements.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" >
 <xs:element name="foo" >
  <xs:complexType>
   <xs:sequence>
<xs:element name="bar" maxOccurs="unbounded" />
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

6

Customizing and Extending Business Objects

Customization and extension are customer side activities. The Oracle Retail release of Functional Artifacts coincides with an Oracle Retail Enterprise Release. Any new Business Objects or changes to existing base Business Objects are handled as described in Chapter 4, "Versioning Artifacts." The packaging schema is discussed in Chapter 2, "Packaging Structure and Standards."

Business Object customization is defined as changes or extension of the base XSDs made by a customer to satisfy a customer specific business requirement.

Oracle Application Integration Architecture (AIA) and other industry standards have defined an approach using well known tags and packaging locations to separate the custom extensions from the base. This allows the extensions to be preserved as updates to the base are applied. Oracle Retail has implemented the same model.

Business Object Customization Tools and Approaches

The *Oracle Retail Functional Artifact Generator Guide* describes the tool used to generate the Functional Artifacts from the Business Object XSDs. The guide also contains examples and step-by-step instructions through simple use cases.

The *Oracle Retail Integration Bus Implementation Guide* contains a chapter about RIB customization and extension, with examples and use cases including customization to the Business Objects.

The Retail SOA Enabler Tool uses localized or custom Functional Artifacts as one of its tool inputs in the process of creating Web services.

Business Object Customization Standards

This section demonstrates the standards used to create and manage functional artifacts, as they relate to the customization and extension of the base business objects (XSDs).

The Functional Artifact Generator tool called the Template Creator is used to create the customization objects and hooks as described in the following example. Using the Template Genertor is recommended for performing the basic steps of creating placeholders and imports.

> **Note:** For a step-by-step example, see the *Oracle Retail Functional Artifact Generator Guide.*

The custom BO schema module must be named ExtOf<BO>.

Topic	Description
Recommendation	The naming convention for the custom XSD for an BO must be as follows: ExtOf<BO>.
Rationale	The naming convention standardizes the names for namespace, packaging, and tool generation of Objects.

Example	ASNInBO.xsd has a corresponding ExtOfASNInBO.xsd:
	
	`xmlns="http://www.oracle.com/retail/integration/base/bo/ASN`
	`InBO/v1"`
	`xmlns:ExtOfASNInBO="http://www.oracle.com/retail/integratio`
	`n/custom/bo/ExtOfASNInBO/v1"`

For a considered custom BO schema, existence of related base BO schema is optional.

Topic	Description
Recommendation	For any BO, existence of a base XSD related to a custom XSD is optional. However the naming convention for the custom XSD must always be as follows: ExtOf<BO>.
Rationale	There can be baseless custom Business Object. Such custom BO do not have related BO.
Example	ExtOfASNInNoBaseBO.xsd does not have a corresponding ASNInNoBaseBO.xsd: /retail/integration/custom/ 　　　bo/ (Business Objects) 　　ExtOfASNInNoBaseBO (Custom BO Name) 　　　v[1] (Version - latest) 　　　　ExtOfASNInNoBaseBO.xsd (BO Schema exist.) /retail/integration/base/ 　　　bo/ (Business Objects) 　　[Business Object ASNInNoBaseBO does not exist]

Each Business Object schema module must import its custom Business Object schema module:

Topic	Description
Recommendation	Each BO schema must import its custom BO schema module.
Rationale	The custom schema is well defined and in the packaging structure.

Topic	Description
| Example | ```
targetNamespace=http://www.oracle.com/retail/integration/base/bo/ASNInBO/v1
 version="1.0"
xmlns="http://www.oracle.com/retail/integration/base/bo/ASNInBO/v1"
xmlns:ExtOfASNInBO=http://www.oracle.com/retail/integration/custom/bo/ExtOfASNInBO/v1
xmlns:retailDoc=http://www.w3.org/2001/XMLSchema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:import

namespace="http://www.oracle.com/retail/integration/custom/bo/ExtOfASNInBO/v1"
schemaLocation="../../../../custom/bo/ExtOfASNInBO/v1/ExtOfASNInBO.xsd">
</xs:import>
``` |

Each BO XSD complexType must have a local element named Custom<Parent Type Name> added.

Topic	Description
Recommendation	Each BO XSD complexType must have a local element named Custom<Parent Type Name> added.
Rationale	A local element named "Custom<Parent Type Name>" exists at the end of every EBO type.
	For example, CustomASNInBO.
| Example | ```
<xs:element maxOccurs="1" minOccurs="0"
ref="ExtOfASNInBO:ExtOfASNInItem">
</xs:element>
``` |

The Custom<Parent Type Name> must not be recursively defined.

Topic	Description
Recommendation	The Custom<Parent Type Name> must not recursively define a note named Custom<Parent Type Name> within it.
Rationale	A Custom<Parent Type Name> must be one level deep, or in relatively flattened structure (to reduce complexity).
Example	

Each BO XSD complexType must have a local element named Localization<Parent Type Name> added.

Topic	Description
Recommendation	The naming convention for the custom XSD for an localized BO must be as follows: EOfBr<BO> or EOfIn<BO>.
Rationale	The naming convention standardizes the names for namespace, packaging, and tool generation of Objects.

Topic	Description

Example	ASNInBO.xsd has a corresponding EOfBrASNInBO.xsd and EOfInASNInBO:
	
	`xmlns="http://www.oracle.com/retail/integration/localizatio n/bo/BrASNInBO/v1"`
	`xmlns:EOfBrASNInBO="http://www.oracle.com/retail/integratio n/custom/bo/EOfBrASNInBO/v1"`

Imports that refer to only non cyclic references are allowed in customized XSD.

Topic	Description
Recommendation	Customized XSD cannot refer to custom localized business objects.
Rationale	Customized business objects can only import non cyclic references.
Example	ExtOfASNOut cannot import EOfBrASNOut.

7

Localizing Retail Business Objects

The deployment of Oracle Retail applications across the globe often drive requirements to localize the messages that flow among the Oracle Retail applications to support the region specific requirements (such as tax laws).

As localization of Business Objects is performed as part of the localization process, where the end-point APIs are changed or added to per region specific business requirements, localizations are meant to be performed by Oracle Retail or Partners. The localization points are not for use by customers. However, localization customization is possible for use by customers.

This chapter demonstrates the standards used to create and manage functional artifacts as they relate to the localization and extension of the base business objects (XSDs).

The Functional Artifact Generator tool called the Template Creator is used to create the localization objects and hooks as described in the following example. The Template Generator is recommended for performing the basic steps of creating placeholders and imports.

> **Note:** For a step-by-step example, see the *Oracle Retail Functional Artifact Generator Guide.*

Business Objects Localization

Business Object localization is defined as extensions of the base XSDS made by Oracle Retail or partners to satisfy region specific business requirements.

Localization Hooks in Base Business Objects

The following table describes the hooks in base Business Objects.

Reference	Description	Example

Every Base BO XSD imports a LocOf\<BO> XSD.	A new placeholder XSD is created for each base Business Object. Every Business Object XSD imports the LocOf\<BO> XSD.	`<xs:import`
		
		`namespace="http://www.oracle.com/retail/integratio`
		`n/custom/bo/ExtOfASNInDesc/v1"`
		`schemaLocation="../../../../custom/bo/ExtOfASNInDe`
		`sc/v1/ExtOfASNInDesc.xsd">`
		`        <retailDoc:annotation>`
		`            <retailDoc:documentation>It's a`
		`referenced element. For detailed description,`
		`please refer referenced element`
		`doc.</retailDoc:documentation>`
		`        </retailDoc:annotation>`
		`    </xs:import>`
		`    <xs:import`
		
		`namespace="http://www.oracle.com/retail/integratio`
		`n/base/bo/LocOfASNInDesc/v1"`
		`schemaLocation="../../../../base/bo/LocOfASNInDesc`
		`/v1/LocOfASNInDesc.xsd">`
		`        <retailDoc:annotation>`
		`            <retailDoc:documentation>It's a`
		`referenced element. For detailed description,`
		`please refer referenced element`
		`doc.</retailDoc:documentation>`
		`        </retailDoc:annotation>`
		`    </xs:import>`

Every Base BO XSD complex type has a localization point.	Within each base Business Object, every complex type contains a reference to a placeholder for that complex type within the LocOf for that Business Object.	`<xs:element name="ASNInItem">`
		`        <retailDoc:annotation>`
		`            <retailDoc:documentation>Description`
		`is not available.</retailDoc:documentation>`
		`        </retailDoc:annotation>`
		`        <xs:complexType>`
		`            <xs:sequence>`
		`            ...`
		`            ...`
		`                <xs:element maxOccurs="1"`
		`minOccurs="0" ref="ExtOfASNInDesc:ExtOfASNInItem">`
		`                    <retailDoc:annotation>`
		
		`<retailDoc:documentation>Provide an extension hook`
		`to customize ASNInItem.</retailDoc:documentation>`
		`                    </retailDoc:annotation>`
		`                </xs:element>`
		`                <xs:element maxOccurs="1"`
		`minOccurs="0" ref="LocOfASNInDesc:LocOfASNInItem">`
		`                    <retailDoc:annotation>`
		
		`<retailDoc:documentation>Provide an extension hook`
		`to localize ASNInItem.</retailDoc:documentation>`
		`                    </retailDoc:annotation>`
		`                </xs:element>`
		`            </xs:sequence>`
		`        </xs:complexType>`

Region Specific Placeholders

The following table describes region specific placeholders.

Reference	Description	Example
LocOfBO imports All Specific Locals	The LocOf XSD is the point at which all specific localization XSDs are imported and referred to. There is one for each region (such as Brazil, India, and China). The ISO standard alpha-2 country code is used as the identifier.	```<xs:schema elementFormDefault="qualified" targetNamespace="http://www.oracle.com/retail/integration/base/bo/LocOfASNInDesc/v1" version="1.0" xmlns="http://www.oracle.com/retail/integration/base/bo/LocOfASNInDesc/v1" xmlns:BrASNInDesc="http://www.oracle.com/retail/integration/localization/bo/BrASNInDesc/v1" xmlns:InASNInDesc="http://www.oracle.com/retail/integration/localization/bo/InASNInDesc/v1" xmlns:retailDoc="http://www.w3.org/2001/XMLSchema" xmlns:xs="http://www.w3.org/2001/XMLSchema"> <retailDoc:annotation> <retailDoc:documentation>This is root element of this document which contains name space definitions for the document elements.</retailDoc:documentation> </retailDoc:annotation> <xs:import namespace="http://www.oracle.com/retail/integration/localization/bo/InASNInDesc/v1" schemaLocation="../../../../localization/bo/InASNInDesc/v1/InASNInDesc.xsd"> <retailDoc:annotation> <retailDoc:documentation>It's a referenced element. For detailed description, please refer referenced element doc.</retailDoc:documentation> </retailDoc:annotation> </xs:import> <xs:import namespace="http://www.oracle.com/retail/integration/localization/bo/BrASNInDesc/v1" schemaLocation="../../../../localization/bo/BrASNInDesc/v1/BrASNInDesc.xsd"> <retailDoc:annotation> <retailDoc:documentation>It's a referenced element. For detailed description, please refer referenced element doc.</retailDoc:documentation> </retailDoc:annotation> </xs:import> </xs:schema>```

Reference	Description	Example
Each of the specific localization XSDs will resolve the references with the actual localization specific elements.	For each of the complex types in the region specific XSD, the place to perform the actual localizations.	```<xs:schema elementFormDefault="qualified"`  `targetNamespace="http://www.oracle.com/retail/integration/localization/bo/BrASNInDesc/v1"` ` version="1.0"`  `xmlns="http://www.oracle.com/retail/integration/localization/bo/BrASNInDesc/v1"` ` xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"` `xmlns:xs="http://www.w3.org/2001/XMLSchema">` ` <xs:import`  `namespace="http://www.oracle.com/retail/integration/custom/bo/EOfBrASNInDesc/v1"` `schemaLocation="../../../../custom/bo/EOfBrASNInDesc/v1/EOfBrASNInDesc.xsd">` ` <retailDoc:annotation>` ` <retailDoc:documentation>It's a referenced element.</retailDoc:documentation>` ` </retailDoc:annotation>` ` </xs:import>` ` <xs:element name="BrASNInItem">` ` <xs:complexType>` ` <xs:sequence>` ` ...` ` ...` ` </xs:sequence>` ` </xs:element>` `</xs:schema>```

Localization Customization

The following table describes the hook for localization customization.

Reference	Description	Example

Each specific localization XSD complex type contains a customiza-tion hook.	In each of the specific localizations, a hook back to the customization of that complex type is referenced.	```xml
		<xs:schema elementFormDefault="qualified"
		
		targetNamespace="http://www.oracle.com/retail/integration/localization/bo/BrASNInDesc/v1"
		    version="1.0"
		
		xmlns="http://www.oracle.com/retail/integration/localization/bo/BrASNInDesc/v1"
		    xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"
		xmlns:xs="http://www.w3.org/2001/XMLSchema">
		    <xs:import
		
		namespace="http://www.oracle.com/retail/integration/custom/EOfBrASNInDesc/v1"
		schemaLocation="../../../../custom/EOfBrASNInDesc/v1/EOfBrASNInDesc.xsd">
		        <retailDoc:annotation>
		            <retailDoc:documentation>It's a referenced element.</retailDoc:documentation>
		        </retailDoc:annotation>
		    </xs:import>
		    <xs:element name="BrASNInItem">
		        <xs:complexType>
		            <xs:sequence>
		              ...
		              ...
		              <xs:element maxOccurs="1" minOccurs="0"
		ref="EOfBrASNInDesc:EOfBrASNInItem">
		                <retailDoc:annotation>
		                  <retailDoc:documentation>Provide an extension hook to customize ASNInItem
		                  </retailDoc:documentation>
		                </retailDoc:annotation>
		              </xs:element>
		            </xs:sequence>
		        </xs:complexType>
		    </xs:element>
		</xs:schema>
		``

Adding Localization Fields

To add the localization specific fields in a placeholder, the first step is to determine the region specific XSD, which follows the naming convention below.

<CC><BO>.xsd (for exampleBrASNInDesc), where CC = 2 char ISO country code.

Determine the complex type fields that must be added. Add the fields and generate payloads using the Artifact Generator tool.

> **Note:** For more information, see the *Oracle Retail Functional Artifact Generator Guide*.

Adding Localization Customization Fields

Customization of localization Business Objects follows the same process required for base Business Objects customization.

8

Importing Retail Business Objects

This chapter provides details about imports that are allowed in each type of retail business object (XSD). A retail business object does not support cyclic reference to another business object.

Imports in Base Business Object (XSD)

A base business object can import another non-cyclic base business object.

Topic	Description
Recommendation	Complex business is composed of small non complex business objects.
Rationale	Base business object can import another base BO that is not a cyclic reference.

Example	`<xs:schema elementFormDefault="qualified" targetNamespace="http://www.oracle.com/retail/integration/base/bo/XItemDesc/v1"  version="1.3"`
	
	 `xmlns="http://www.oracle.com/retail/integration/base/bo/XItem/v1"`
	
	 `xmlns:XItmSupCtyMfr=http://www.oracle.com/retail/integration/base/bo/XItmSupCtyMfr/v1 >`
	
	 `<xs:import namespace="http://www.oracle.com/retail/integration/base/bo/XItmSupCtyMfr/v1" schemaLocation="../../XItmSupCtyMfr/v1/XItmSupCtyMfr.xsd">`
	 `</xs:import>`

A base business object must import generic localized business object.

Topic	Description
Recommendation	A base business object <BO>, must provision Localization by importing generic localized business object LocOf<BO>.
Rationale	A base business object must import generic localized business object LocOf<BO>.

Topic	Description

Example	`<xs:schema elementFormDefault="qualified"`
	
	`targetNamespace="http://www.oracle.com/retail/integration/base/bo/XItem/v1"`
	`    version="1.3"`
	
	`xmlns:LocOfXItem="http://www.oracle.com/retail/integration/base/bo/LocOfXItem/v1"`
	
	`<xs:import`
	`namespace="http://www.oracle.com/retail/integration/base/bo/LocOfXItem/v1"`
	`schemaLocation="../../../../base/bo/LocOfXItem/v1/LocOfXItem.xsd">`
	`</xs:import>`

Imports in Custom Business Object (XSD)

A custom business object can import non-cyclic child custom business objects.

Topic	Description
Recommendation	A custom business object can be composed of child custom business objects.
Rationale	A custom business object, ExtOf<BO> can import custom child business objects, ExtOf<Child Object1> , ExtOf<Child Object2>.

Example	`<xs:schema elementFormDefault="qualified"`
	
	`targetNamespace="http://www.oracle.com/retail/integration/base/bo/ XItmSupCtyMfr /v1"`
	`version="1.3"`
	
	`xmlns="http://www.oracle.com/retail/integration/base/bo/ XItmSupCtyMfr /v1"`
	
	`xmlns:ExtOfXItmSupCtyMfr=http://www.oracle.com/retail/integration/base/bo/ ExtOfXItmSupCtyMfr /v1`
	
	`>`
	`<xs:import`
	
	`namespace="http://www.oracle.com/retail/integration/base/bo/`
	
	`ExtOfXItmSupCtyMfr /v1" schemaLocation="../../ `
	
	`ExtOfXItmSupCtyMfr /v1/ ExtOfXItmSupCtyMfr.xsd">`
	
	`</xs:import>`

Imports in Localized Business Object (XSD)

A localized business object can import localized child business objects for the same region only.

Topic	Description
Recommendation	A localized business object can compose of localized child business objects with same country code only.
Rationale	A localized business object, In<BO> can import custom localized business objects In<child BO> with same country code.

Example	```
	<xs:schema elementFormDefault="qualified"
	
	targetNamespace="http://www.oracle.com/retail/integration/localization/bo/InSupplier/v1"
	    version="1.0"
	
	xmlns="http://www.oracle.com/retail/integration/localization/bo/InSupplier/v1"
	
	xmlns:InSupplierDtl="http://www.oracle.com/retail/integration/custom/bo/ InSupplierDtl /v1"
	    xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"
	xmlns:xs="http://www.w3.org/2001/XMLSchema">
	    <xs:import
	
	namespace="http://www.oracle.com/retail/integration/custom/bo/EOfInSupplierDesc/v1"
	schemaLocation="../../../../custom/bo/ InSupplierDtl /v1/InSupplierDtl.xsd">
	</xs:import>
	``

A localized business object can import custom localized business objects for the same region only.

Topic	Description
Recommendation	A localized business object can compose of customized localized business objects with same country code only.
Rationale	A localized business object, In<BO> can import custom localized business objects EOfIn<BO> with same country code.

Topic	Description
| Example | ```xml
<xs:schema elementFormDefault="qualified"

targetNamespace="http://www.oracle.com/retail/integration/localization/bo/InSupplier/v1"
    version="1.0"

xmlns="http://www.oracle.com/retail/integration/localization/bo/InSupplier/v1"

xmlns:EOfInSupplierDesc="http://www.oracle.com/retail/integration/custom/bo/EOfInSupplier/v1"
    xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:import

namespace="http://www.oracle.com/retail/integration/custom/bo/EOfInSupplierDesc/v1"
schemaLocation="../../../../custom/bo/EOfInSupplierDesc/v1/EOfInSupplier.xsd">
        <retailDoc:annotation>
            <retailDoc:documentation>It's a referenced element. For detailed description, please refer referenced element doc.</retailDoc:documentation>
        </retailDoc:annotation>
    </xs:import>
``` |

A

References

The following is a list of reference materials providing more information on the topics covered in this guide.

- [XML] Extensible Markup Language (XML) 1.0 (Second Edition)
  http://www.w3.org/TR/REC-xml

- [XMLSchema1] W3C Recommendation, XML Schema Part 1: Structures
  http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

- [XMLSchema2] W3C Recommendation, XML Schema Part 2: Datatypes

  http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

- [OAGINDR] OAGi OAGIS 9.0 Naming and Design Rules Standard version 0.7, published September 30, 2005

- [CCTS] UN/CEFACT Core Components Technical Specification Draft 2.2, published March 31 ,2006.

  www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf

- [UNCEFACTNDR] UN/CEFACT XML Naming and Design Rules, Draft 2.0, published January 20, 2006

  http://www.unece.org/cefact/xml/xml_index.htm

- Oracle Application Integration Architecture - Enterprise Object Library 2.2: Enterprise Business Objects and Messages XML Naming and Design Rules Part No. E12769-01 August 2008.

- ARTS SOA Best Practices Technical Report Version 1.2.0 November 19, 2008 - Candidate Recommendation.

- "Web Service Contract Design and Versioning for SOA," Thomas Erl, et. al., September 2008, Prentice Hall.

- WS-I Basic Profile 1.1.
  http://www.ws-i.org/Profiles/BasicProfile-1.1.html

Glossary

The following is a list of acronyms used within this guide.

ABCS

Application Business Connector Service

AIA

Application Integration Architecture

ABM

Application Business Message

ABO

Application Business Object

BO

Retail Business Object

EBM

Enterprise Business Message

EBO

Enterprise Business Object

ESB

Enterprise Service Bus

ODI

Oracle Data Integrator

POJO

Plain Old Java Object

RGBU

Retail Global Business Unit

RTG

Retail Technology Group

SOA

Service Oriented Architecture: This term refers to an architectural style of building reliable distributed systems that deliver functionality as services, with additional emphasis on loose coupling between interacting services.