Oracle® Transportation Management

Integration Guide

Release 6.3

Part No. E38428-08

September 2015

ORACLE

Oracle Transportation Management Integration Guide, Version 6.3

Part No. E38428-08

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 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.

Contents

Table of Figures

Send Us Your Comments

Oracle Transportation Management Integration Guide, Release 6.3

Part No. E38428-08

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: otm-doc_us@oracle.com

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, contact Support at https://support.oracle.com or find the Support phone number for your region at http://www.oracle.com/support/contact.html.

Preface

This manual is for members of the Oracle Transportation Management and Global Trade Management implementation teams, who are responsible for connecting the system to other external systems through integration interfaces. This manual explains how to send and receive integration messages and the format of each message.

This manual does not cover the installation of any components required to import or export. See the Administration Guide and Installation Guide in your Oracle Transportation Management installer for installation and configuration instructions.

Change History

Date	Document Revision	Summary of Changes
12/2012	-01	Initial release. Remove redundant information and improve layout. Enhanced Web Service Security support. Remove references to ReleaseType (BUG14531888) Deprecation of LargeTransmissionServlet (BUG14588410) New DemurrageTransaction, GtmShipment and GtmRegistration interfaces.
02/2013	-02	Removed glog.integration.servlet.WMServletClient from chapter 3.
07/2013	-03	Modified explanation of Internal Processing.
11/2013	-04	Added note to consider use of GenericStatusUpdate in preference to ActualShipment for simple modifications. Moved section on Oracle Advanced Queues to here from Data Management Guide. Included description of Composite Executor for Direct Insert XML Data Queues.
05/2014	-05	Added the section "Adding Stops" to chapter 7. Updated the "Processing Global Trade Management Content" section to better reflect active processes. Specify supported `Content-Type` values for HTTP POST messages.
08/2014	-06	Added DIR_XML_USER information in chapter "Direct XML Insert"
07/2015	-07	The quote interface is supported on the outbound only.

Date	Document Revision	Summary of Changes
09/2015	-08	Added description of new capability to specify stylesheet content GID instead of stylesheet name.

1. Integration Overview

The Oracle Transportation Management (OTM) and Global Trade Management (GTM) suite of products use XML messages to interface with external applications.

OTM and GTM have separate interfaces for processing different messages. For example, the Transportation Orders (`TransOrder`) interface receives transportation order messages from an external application into OTM. This is referred to as an **inbound** interface. The **outbound** interfaces send messages from the OTM/GTM application to other external applications. For example, shipments planned from Orders in OTM can be sent in a `PlannedShipment` message to another system for additional processing.

The valid formats of all messages are described by XML Schema Definitions (XSD) documents.

This document will describe the following:-

- The purpose of each XSD and the available messages defined by it.
- The application level message protocol describing the process of message exchange.
- The transport level protocols supported for inbound and outbound message communication.

2. Understanding the Integration Schemas

This section will describe the overall format and design of the Integration XML Schema Definition (XSD) schemas for use when implementing interfaces.

Definitions

The following definitions are used widely in the remainder of the document and deserve detailed explanation of their exact meaning.

Term	Definition
Global element/type/attribute	This is an XSD definition which is an immediate child of the <xsd:schema> document element.
Local element/type/attribute	This is an XSD definition that is contained within a parent definition e.g. an element definition within a named complex type.
Local reference	This refers to using the 'ref=' attribute to reference a global element definition i.e. as opposed to a Local element definition.

For mainly historical reasons, most element definitions are defined as Global element definitions and are referenced from local references within other element or type definitions.

This is in contrast to the current best practice where Global element definitions should only be used to define XML fragments that can exist sensibly as isolated documents (referred to as 'Primary Documents' in this documentation) or as targets for substitution groups. Therefore it is important to read and understand the following section – Primary XML Documents – which will identify and describe those Global elements intended to be used as XML documents for interface messaging.

The OTM/GTM XML Schemas

The OTM/GTM XML schemas define the data elements that the system sends or receives for each type of interface. The XML schema definitions are considered the true definition for the interfaces, and this Integration Guide is the supporting documentation for the schemas. Information appearing in the schemas overrides information in this guide if there is a conflict.

The GLogXML schema defines the Transmission related messages and the interface transaction formats, contained in the Transmission, which would be sent to Oracle Transportation Management.

> **Note:** Both the XML schema and the online help describes each element. The XML Element Dictionary, which is distributed with documentation, contains definitions of each element.

The GLogXML-GTM schema describes the interface transaction formats, also contained in a Transmission, but which would be sent to Oracle Global Trade Management.

The Mobile Message Schema defines the format in which you send mobile device messages.

The Common Schema defines some common data types that are currently only used by the Message Schema but are intended for future use.

W3C XML Schema Version

All schema files conform to the W3C XML Schema standard (see http://www.w3.org/XML/Schema) defined by the following namespace name:

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

XML Schema Namespaces

Each XML schema is defined with a specific target namespace. The following table lists the current target namespaces and associated physical file name for each schema:

Schema	XSD Schema File	Namespace
GLogXML Schema	GLogXML.xsd	http://xmlns.oracle.com/apps/otm
Mobile Message Schema	Message.xsd	http://xmlns.oracle.com/apps/otm/msg/v1
GLogXML-GTM Schema	GLogXML-GTM.xsd	http://xmlns.oracle.com/apps/gtm
Common Schema	Common.xsd	http://xmlns.oracle.com/apps/otm/types

All XML transmissions sent inbound to Oracle Transportation Management SHOULD correctly use namespace names as defined by the XML schemas listed above. However, for backward compatibility, interface transmissions are not REQUIRED to specify the namespace in the XML (except when using inbound web services; see section 12 Oracle Transportation Management Web Services). Please note that this support is DEPRECATED and will be removed in the next major release of Oracle Transportation Management after which time valid namespace names will be REQUIRED.

When XML documents are sent outbound from Oracle Transportation Management, the namespace attribute is specified by default. See Section: Excluding Namespace Names in Outbound Interfaces for details on how this can be configured.

XML Schema Changes

All changes to XSD documents for a particular release are described in the XML Interface Changes Guide document.

Viewing the XML Schemas

The schema files can be obtained from the following User Interface menu location:-

Business Process Automation > Integration > Integration Manager > Retrieve Schemas

When integrating to Oracle Transportation Management using XML, you must create documents that follow the structure and rules of the Oracle Transportation Management XML schemas. We recommend that you use an XML management tool to view the schema files. This will help in understanding the Oracle Transportation Management and Global Trade Management data elements and relationships.

The W3C XML Schema site (http://www.w3.org/XML/Schema) provides links to several such tools. The examples in this document use the freely available Oracle XSD Visual Editor which is a built-in part of the JDeveloper IDE (see http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html for details.)

For a full description of the Visual Editor tool please see Developing Applications Using XML in the Oracle Fusion Middleware User's Guide for JDeveloper online documentation (http://docs.oracle.com).

Common Schema Elements and Types

Global IDs (GIDs)

GIDs are global identifiers that Oracle Transportation Management uses to define various types of information (e.g., orders, shipments, locations, payment vouchers, etc.). A GID consists of the following two parts:

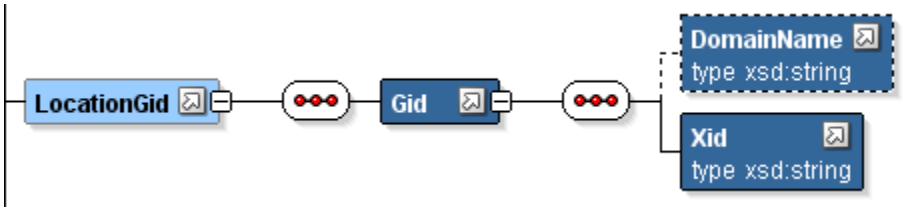
- **Domain name**: Typically identifies a company and is used to separate data and secure it from other data in a shared, web-based environment. For example, if you are using Oracle Transportation Management in an environment where many companies may be using the same Oracle Transportation Management installation, the domain allows you to isolate data in Oracle Transportation Management for each company. Therefore, many users from different companies can work in the same Oracle Transportation Management installation (or web site) and use data that is private and specific to their company. If you do not include a domain name in a GID, it can be viewed across all domains in your system.
- **External ID (XID)**: The ID that defines the item on the external system. An external system is any system other than Oracle Transportation Management.

  Note: You should not create GIDs with trailing spaces, as these records will not be able to be looked up via the UI.

The GLogXML and GLogXML-GTM schemas use a containment model for specifying instances of GID data types whereas the Message schema uses a more concise reference type. The former design was due in most part to concerns of interoperability as it did not depend on sophisticated (at the time) use of XSD schemas. It is anticipated that in the near future the GLogXML and GLogXML-GTM schemas will migrate to also use a reference type.

Containment Model example

The following is an example of a GID – LocationGid – which uses containment.

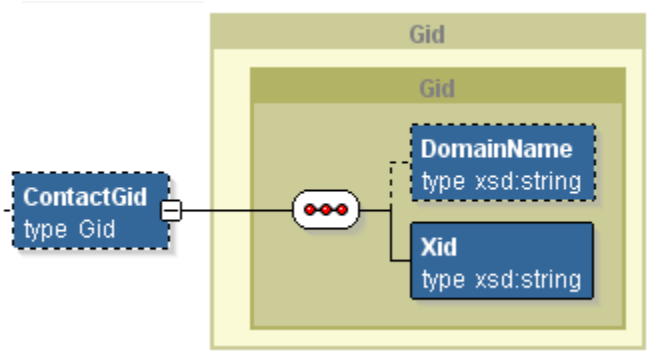


A valid LocationGid XML element would look like the following:-

```
<?xml version="1.0" encoding="utf-8" ?>
<LocationGid>
    <Gid>
        <DomainName>GUEST</DomainName>
        <Xid>ABCD</Xid>
    </Gid>
</LocationGid>
```


Complex Type Reference example

The following is an example from the Message.xsd schema which uses a referenced type[1].



A valid ContactGid XML element in a Mobile Device Message XML would look like the following:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContactGid>
    <DomainName>GUEST</DomainName>
    <Xid>XYZ</Xid>
</ContactGid>
```

IMPORTANT NOTE: As can be seen, the referenced type results in a more concise representation of GID data. Until such time as the GLogXML and GLogXML-GTM schemas are enhanced to use the reference type, care must be taken when working with interface messages of both types to make sure the correct model – containment or reference type – is used.

Date & Time

The GLogDateTimeType schema type is a platform neutral representation of the date and time to the accuracy of seconds. It is a string in the format YYYYMMDDHHMMSS where:

YYYY	-	The year e.g. 2012
MM	-	The month e.g. 01 is January, 12 is December
DD	-	The day number e.g. 01 to 31
HHMMSS	-	The time in 24 hour clock e.g. 231530

An example value is "20121031083030" which represents October 31st, 2012 at 30 seconds after 8.30a.m.

---

[1] This actually shows that the GID type in Message.xsd extends the GID type in another schema (Common.xsd). This merely a mechanism to force the element to be defined in the Message.xsd target namespace and does not affect the XML format.

Additionally, the GLogDateTimeType can hold a Time Zone ID element (TZId) and a Time Zone Offset. When date elements are inbound the Time Zone Id is used to calculate times relative to the system time. On outbound messages, the Time Zone Offset will be populated to show the offset in hours from the system Time Zone compared to the target system locale. See Time Zone in Integration XML section for details.

Flex Fields

Flex fields are so called because they provide a flexible of table columns for OTM/GTM implementers to customize their data model to suit their functional needs. They can be entered via the User Interface and/or via Integration.

Flex fields cover a number of data types and have been added to most of the major interfaces and are available for both inbound and outbound messages.

The valid types are described in the following table

Schema Type	DB Data type	Number	Description
FlexFieldStringType	VARCHAR2(150)	20	Holds string data
FlexFieldNumberType	NUMBER()	10	Holds numbers with any precision
FlexFieldCurrencyType	OTM Currency UOM	3	Holds financial amount
FlexFieldDateType	DATE	10	Holds date/time values

Not all flex field types have been added to all interfaces. The additions are outlined in the following table

Element	String	Number	Currency	Date
Release/ReleaseHeader	x	x	x	x
Item	x	x		x
ShipmentStatus	x	x		x
ShipUnit	x	x		x
TransOrder/TransOrderHeader	x	x	x	x
TransOrder/TransOrderLine	x	x		x
Location	x	x		x
Shipment/ShipmentHeader	x	x	x	x
Equipment	x	x		x
ShipmentStop	x	x		x

Element	String	Number	Currency	Date
PaymentHeader	x	x	x	x
Voucher	x	x	x	x
Packaging	x	x		x
ShipmentGroup/ShipmentGroupHeader	x	x		x
Job	x	x		x
Consol	x	x		x
Claim	x	x	x	x
QuoteHdr	x	x		x
Driver	x	x		x
PowerUnit/PowerunitHeader	x	x		x
Device	x	x		x
OrderMovement	x	x		x
Itinerary	x	x		x

In addition to the above, the RATE_GEO and RATE_OFFERING outbound interfaces have equivalent Flex Fields available which are defined slightly differently due to the database centric nature of these (outbound only) interfaces. These interfaces use the following elements:-

Schema Element	DB Data type	Number	Description
ATTRIBUTE1…n	VARCHAR2(150)	20	Holds string data
ATTRIBUTE_NUMBER1…n	NUMBER()	10	Holds numbers with any precision
ATTRIBUTE_DATE1…n	DATE	10	Holds date/time values

Primary XML Documents

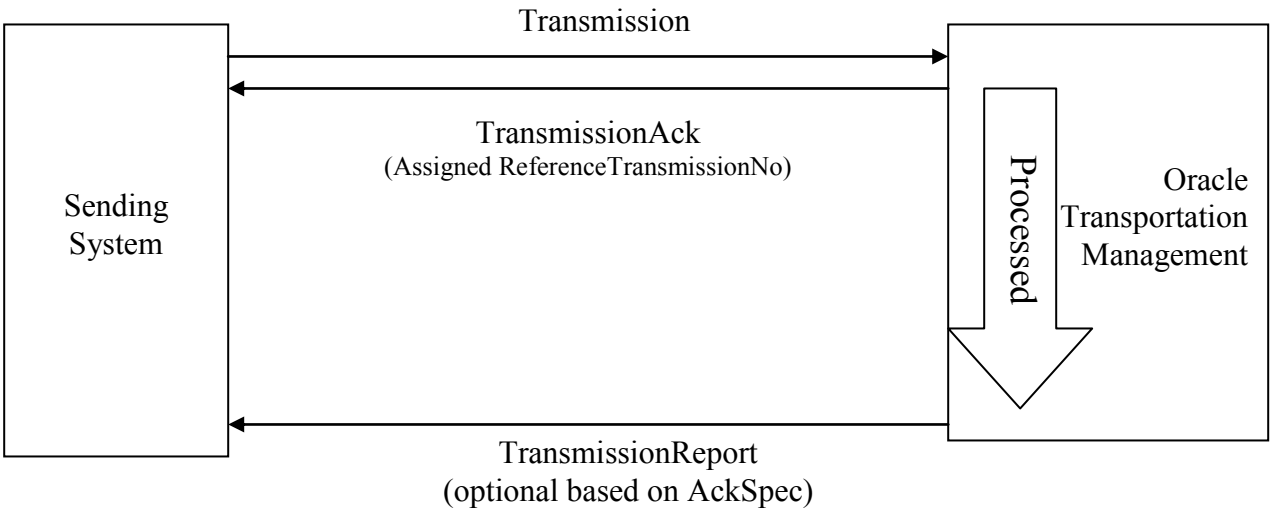
GLogXML Schema

There are four primary XML documents defined in the GLogXML schema that are used inbound and outbound for both Oracle Transportation Management interfaces (also defined in GLogXML) and Global Trade Management interfaces (defined in GLogXML-GTM):

- Transmission

    - The Transmission is the primary document used for inbound to and outbound from the system. Each Transmission can contain multiple transactions to be processed.
- TransmissionAck
    - The TransmissionAck is the response message to the receipt of the Transmission. It contains the confirmation for the receipt of the Transmission with an assigned ReferenceTransmissionNo element, or an error if the Transmission was not correctly received.
- TransmissionReport
    - The TransmissionReport summarizes the errors that were detected during the processing of the Transmission. The report is optionally sent after all the transactions in the Transmission have been completed (successfully processed or generated errors). The request for the TransmissionReport is indicated in the Transmission in the AckSpec element in the TransmissionHeader.
- TransactionAck
    - The TransactionAck is supported inbound to the system as an acknowledgement of transactions that have been sent outbound from the system.

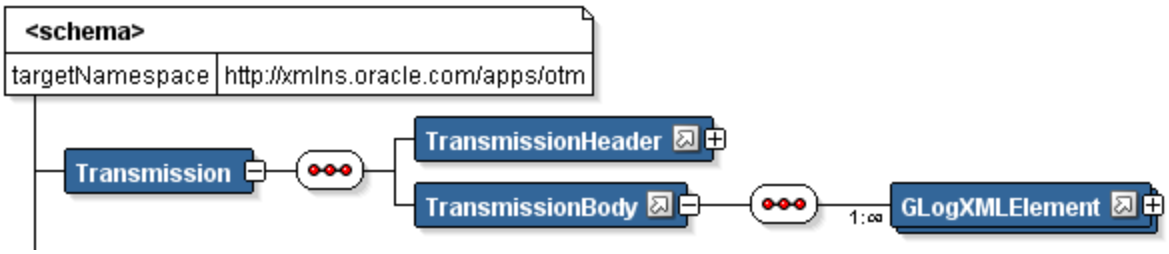
Each of these documents is detailed in the GLogXML schema file. The following diagram shows when the documents are sent for an inbound scenario into Oracle Transportation Management.



Transmission Structure

The Transmission XML document has a root element with local name `Transmission`. The current namespace name is `http://xmlns.oracle.com/apps/otm`.

The `Transmission` contains a `TransmissionHeader` and a `TransmissionBody`. The `TransmissionBody` contains one or more interface transactions wrapped in a `GLogXMLElement` element. See figure below for schema diagram representation.



The following is an example XML containing two transaction interfaces (`TransOrder` followed by `ShipmentStatus`).

```
<ns1:Transmission xmlns:ns1="http://xmlns.oracle.com/apps/otm">
    <ns1:TransmissionHeader>
        <ns1:SenderTransmissionNo>1234-ABCD</ns1:SenderTransmissionNo>
        <ns1:AckSpec>
            <ns1:ComMethodGid>
                <ns1:Gid>
                    <ns1:Xid>HTTPPOST</ns1:Xid>
                </ns1:Gid>
            </ns1:ComMethodGid>
            <ns1:ServletURL>http://myserver/com.example.TransmissionReportReceiver</ns1:ServletURL>
            <ns1:AckOption>ERROR</ns1:AckOption>
        </ns1:AckSpec>
    </ns1:TransmissionHeader>
    <ns1:TransmissionBody>
        <ns1:GLogXMLElement>
            <ns1:TransOrder></ns1:TransOrder>
        </ns1:GLogXMLElement>
        <ns1:GLogXMLElement>
            <ns1:ShipmentStatus></ns1:ShipmentStatus>
        </ns1:GLogXMLElement>
    </ns1:TransmissionBody>
</ns1:Transmission>
```

AckSpec

The AckSpec element controls whether or not a TransmissionReport message is sent when the transactions within a Transmission have been completely processed i.e. when they have been persisted and any dependent workflow has been completed.

The AckOption elements let you specify when to receive a TransmissionReport. If unspecified, the behavior is property controlled where the default property setting (glog.integration.TransmissionReport=”on error”) is equivalent to the `ERROR` option.

- ERROR = Send Transmission Report only when there are errors.
- YES = Send Transmission Report in all cases.
- NO = Do not send Transmission Report, even if there are errors.

Mobile Message Schema

The Message XML is used for mobile communication messages sent inbound and outbound to Oracle Transportation Management. These messages are intended for communication to a mobile communication device or sensor. When sent inbound to Oracle Transportation Management, these messages should be small. The message body can contain text-formatted content, which may be parsed and converted to the Transmission XML when transmission processing is required. There are

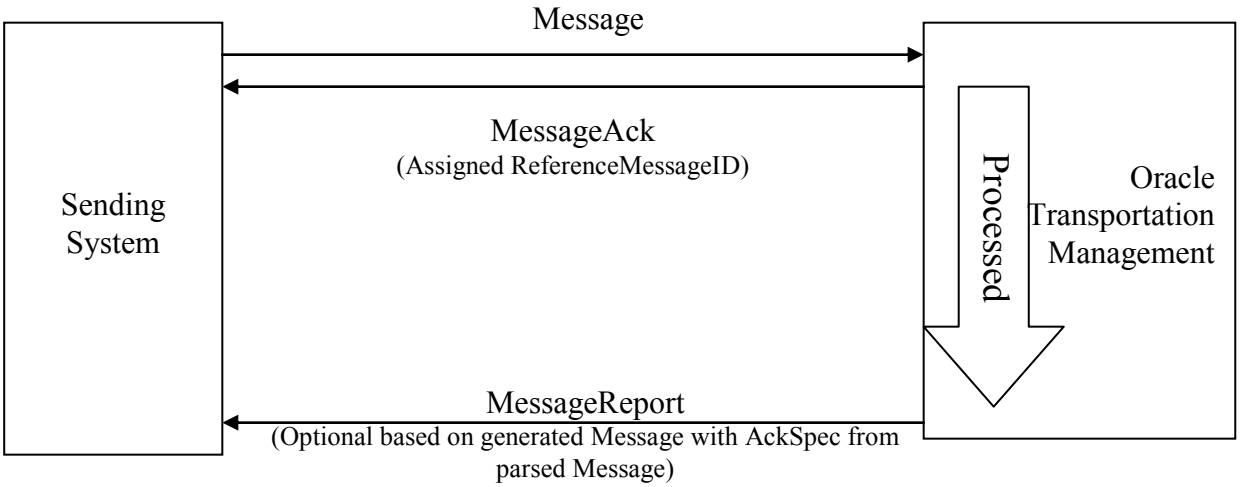
two primary XML documents in the Message XML schema that are used inbound and outbound to Oracle Transportation Management:

- Message
    - The Message is the primary document used for inbound to and outbound for mobile communication messages. Each message can contain either text based or XML-based content in the message body.
- MessageAck
    - The MessageAck is the response message to the receipt of the message. It contains the confirmation for the receipt of the message with an assigned ReferenceMessageId element, or an error if the message was not correctly received.
- MessageReport
    - The MessageReport summarizes the errors that were detected during the processing of the message. The report is optionally sent after all the transactions in the Transmission have been completed (successfully processed or generated errors). The request for the MessageReport is indicated in the message in the AckSpec element in the MessageHeader. The MessageReportBody in the MessageReport will include the TransmissionReport that will summarize the errors that were detected during the processing of the Transmission.

Each of these documents is detailed in the Message XML schema file.

In the case where the message is correctly parsed and converted to a Transmission XML for inbound transmission processing, the GLog TransmissionReport XML may be sent as a summary of processing if the AckSpec is specified in the generated Transmission resulting from the parsing.

The following diagram shows when the documents are sent for an inbound scenario into Oracle Transportation Management.



3. Send Data Inbound

There are various ways to send integration transmissions to Oracle Transportation Management:

- Send via HTTPPOST to one of the following servlets on the web server:
    - http://hostname/GC3/glog.integration.servlet.WMServlet
        - WMServlet is the default servlet used when sending the Transmission or Message XML.
    - http://hostname/GC3/glog.integration.servlet.LargeTransmissionServlet
      DEPRECATED: Please instead use WMServlet described above.
        - LargeTransmissionServlet can be used for sending exceptionally large Transmission(s) into Oracle Transportation Management. The difference with WMServlet is that parsing of the XML is handled in the servlet, and there is suppression of storing the complete Transmission in the database. The individual transactions are stored in the database.
    - http://hostname/GC3/glog.integration.servlet.TransformerServlet
        - TransformerServlet is used to apply an XSL transformation to an XML to convert it into a valid Transmission XML. Refer to the **Transform Inbound XML with XSL** section for additional details.
    - http://hostname/GC3/glog.integration.servlet.DirLoadServlet
        - DirLoadServlet provides a faster option than WMServlet for loading data into Oracle Transportation Management by bypassing the application server. It can be used for inserting/creating data. Refer to the **Send Data and Load Faster into Oracle Transportation Management** section for additional details.

  Note: The supported content types for HTTP messages (corresponding to the `Content-Type` HTTP header parameter) are `text/xml` and `application/xml`.

- Send data via a web service to the application tier.
- Use Oracle Advanced Queuing to send XML transmissions to Oracle Transportation Management. See section 9 for details.

  Note: This is currently not supported for the Message XML in Oracle Transportation Management version 6.3.

- Manually upload an XML file in the Integration Manager. This is primarily used for testing.
- Directly insert Transmission XML into Oracle Transportation Management database. See the **Direct XML Insert** section for details.

User Authentication for Inbound Transmissions

Oracle Transportation Management requires authentication when sending in the Transmission or Message XML. The following options are available:

- The username and password can be specified in the Transmission XML via the UserName and Password elements in the TransmissionHeader element. For the Message XML, the username and password can be specified via the Username and Password elements.
- When sending via HTTP post to one of the servlets, the username and password can be specified in the HTTP header.
- When sending via HTTP post to one of the servlets, the username can be specified with no password in the HTTP header, and the IP address of the sending system can be used for validation. The IP address would need to be specified in the External System Manager in the UI. Refer to online help for additional details.

- When sending via Oracle Advanced Queue, the username and password can be specified as elements in the INTG_QUEUE_MESSAGE message payload type.
- When sending via Direct XML Insert, the username and password are specified as parameters on the PL/SQL procedure. See Section 11, Direct XML Insert for details.
- When sending via a web service, the username and password can be specified in the Web Service Security Username Token. See **Oracle Transportation Management Web Services** for more details.

Password Encryption

IMPORTANT: In versions of OTM & GTM prior to version 6.3, the password element in the Transmission and Message XML messages could be specified as plain text or encrypted.

However, due to increased security related to password protection, this capability is now no longer supported and so, if a password is used, only plain text passwords are supported.

Therefore, as is common security practice, it is recommended to only send plain text passwords when the network connection between the external server and Oracle Transportation Management can be guaranteed as secure, for example, over HTTPS.

Enable Parsing in Oracle Transportation Management Servlets for Namespaces

With the addition of the namespace to the Oracle Transportation Management schema, parsers may need to be enabled in the Oracle Transportation Management web servers to deal with namespaces in the elements. For example, an inbound Transmission with a namespace and namespace prefixes for all elements could appear as:

```
<Transmission xmlns="http://xmlns.oracle.com/apps/otm">
<ns1:TransmissionHeader xmlns:ns1="http://xmlns.oracle.com/apps/otm">
<ns1:UserName>MYDOMAIN.ADMIN</ns1:UserName>
<ns1:Password>MYPASSWORD</ns1:Password>
</ns1:TransmissionHeader>
<ns1:TransmissionBody xmlns:ns1="http://xmlns.oracle.com/apps/otm">
…
<ns1:TransmissionBody>
</Transmission>
```

This would need to be parsed within Oracle Transportation Management to correctly extract the data (e.g. the values for the username and password used for authentication).

The parsers within the application can be enabled through setting the following property:

```
glog.integration.enableParserInServlets=true
```

Send Data and Load Faster into Oracle Transportation Management

You can get Oracle Transportation Management to load your inbound transmissions into the database faster than the WMServlet and without involving the application server. This is good when you just want to load data into Oracle Transportation Management and process the data later, like during setup of Oracle Transportation Management.

DirLoadServlet only supports these interfaces:

- TransOrder: You must include the GID to be able to have the application server offline.
- Shipment
- ShipmentLink
- TenderResponse
- Location
- Item
- ItemMaster
- HazmatGeneric
- HazmatItem
- ShipmentStatus
- Invoice
- Release
- ShipmentGroup
- SShipUnit
- Sku
- SkuTransaction
- Contact
- TransOrderStatus

  Note: The DirLoadServlet does not raise events (like shipment - created for ActualShipment) so automation agents cannot be triggered. This means that to realize the benefits of the DirLoadServlet, you should only use interfaces that do not require processing. Examples of interfaces not requiring processing are Location, Contact, and Item. See the agent manager online help to learn more about what agents cannot start when using DirLoadServlet.

  Note: Oracle Transportation Management ignores your AckSpec element. Instead, the DirLoadServlet HTTPPOSTs the TransmissionAck back to the IP address you sent your Transmission from.

To do this:

1. Make sure your transmissions only use the transaction code I.

   It is possible to use other transaction codes but with the limitation that you need to make sure that no user accesses that data through the application server while you update/delete your data. If a user accesses the data, you need to restart your application server after uploading your data to refresh its caches. To use other transaction codes with the DirLoadServlet you need to enable them in glog.properties.
2. If you load many transactions and want to increase loading speed, you can increase the number of threads assigned to load the data in glog.properties.
3. Post XML transmissions to http://hostname/GC3/glog.integration.servlet.DirLoadServlet

Result

1. DirLoadServlet saves your data to the database.
2. DirLoadServlet sets default statuses for business objects you insert.

Send Data via a Web Service to the Application Tier

You can send data to Oracle Transportation Management via a web service call to the Oracle Transportation Management application server. If your system is configured as a cluster of more than one application server, each application server will provide a web service endpoint. At the current time, each application server has to be accessed directly and so any load balancing or failover across application servers would have to be managed external to the OTM/GTM system e.g. by a hardware load balancer.

Servlet for Accessing Web Services Description Language (WSDL) Files

Each application server provides a different service endpoint URL for accessing the WSDL file. As a convenience, a new servlet has been added to Oracle Transportation Management that correctly points to the URL for the WSDL based on the application server. The servlet is accessed via the following menu path: Business **Process Automation > Integration > Integration Manager > Retrieve WSDLs** on main page. The servlet relies on the following properties to form the correct URL:

- appserver
- appserver.port.webservice.weblogic (default value is 7001)

If the application server configures a different port for use, please add the properties with the appropriate values in your glog.properties file.

The servlet also provides access to any related XSD schema files for each service.

Available Services on WebLogic and Oracle Application Server Platforms

The following web services are available for inbound processing:

- **IntXmlService**: The IntXmlService is the name of the supplied integration web service for receiving the Transmission XML. You can retrieve the WSDL file for the service as described above. It is a document style service. The operation is as follows:
    - process: accepts a Transmission XML document and responds with the TransmissionAck xml. When the Transmission is for a query as in the case of Rate Inquiry (RIQQuery), the results of the query are embedded in the QueryResultInfo element in the TransmissionAck element. Note that this is a change from previous responses where the RemoteQueryReply or the Transmission elements were returned for query responses.
- **IntGtmXmlService**: The IntGtmXmlService is the name of the supplied integration web service for receiving the Global Trade Management XML interfaces. You can retrieve the WSDL for the service as described above. It is a document style service.
    - One issue identified in JDeveloper when working with the IntGtmXmlService is that JDeveloper had an issue with allowing you to select the Global Trade Management elements from the Global Trade Management schema. A workaround is to manually edit the business process execution language (BPEL) file as follows:
        1. Add the namespace to the root element after the Oracle Transportation Management namespace attribute. For example:

```
<process name="SendGtmParty"
         ...
          xmlns:ns2="http://xmlns.oracle.com/apps/otm"
          xmlns:ns3="http://xmlns.oracle.com/apps/gtm">
```


2. Add the variable for the Global Trade Management interface needed. For example, if you need the GtmParty interface, add the following variable noting the namespace prefix for the Global Trade Management schema as identified in step 1 above:

```
    <variables>
        <variable name="transmission"
element="ns2:Transmission"/>
        <variable name="gtmParty" element="ns3:GtmParty"/>
    </variables>
```

- After these two changes are manually made, you can proceed with developing the BPEL flow.

- **MessageService**: The MessageService is the name of the supplied integration web service for receiving the Mobile Message XML. You can retrieve the WSDL file for the service as described above. It is a document style service.

Transform Inbound XML with XSL

You can get Oracle Transportation Management to transform your inbound transmissions from another XML schema to GLogXML. To do this:

1. Upload the XSL file or files that Oracle Transportation Management must use to transform your inbound transmissions. This can be done by uploading the file via the Integration Manager UI. This activity must be performed as a user with DBA.ADMIN privileges. Alternatively, a Stylesheet Content record can be created in the database which will then contain the uploaded XSLT content.
2. Include this processing instruction in the beginning of every transmission that Oracle Transportation Management needs to transform:

   ```
   <?gc3-int-translate stylesheet_name=”stylesheet_name”?>
   ```

   stylesheet_name is the file name of the XSL file you have uploaded or the stylesheet content GID if the file has been uploaded previously to the database. If using the stylesheet content, you must prefix the GID with the string “db:”. You can include multiple processing instructions in one transmission, in which case, Oracle Transportation Management will transform in the order the processing instructions appear in the transmission.
3. Post XML documents to http://hostname/GC3/glog.integration.servlet.TransformerServlet instead of http://hostname/GC3/glog.integration.servlet.WMServlet

Transaction Codes

Transaction codes tell Oracle Transportation Management what to do with the transmissions it receives from other systems. In the TransactionCode element field for each of the interfaces, enter one of the following values:

- I: Insert. Use this transaction code to send new information to Oracle Transportation Management. Oracle Transportation Management creates a new record. If the record already exists, then the transaction will generate a "record already exists" error.
- II: Insert and Ignore. When used, if the record already exists, then it is not updated and the "record already exists" error message is not logged. If it does not exist, then it is inserted.
- U: Update. Oracle Transportation Management updates an existing record.
- UU: Oracle Transportation Management updates the existing record while suppressing “no data found” constraint violations.

- IU: Insert and Update. Oracle Transportation Management creates a new record unless it already exists, in which case Oracle Transportation Management updates the existing record with the new information.
- UI: Update and Insert. This works the same way as IU.
- D: Delete. Use this transaction code to delete an existing record.
- DD: Oracle Transportation Management deletes the existing record while suppressing "no data found" constraint violations.
- NP: No Persist. Use this transaction code to keep Oracle Transportation Management from persisting data to the database. For example, enter NP if you do not want to persist public locations. This is the default TransactionCode when the data is sent outbound from Oracle Transportation Management.
- RC: Replace Children. Use this transaction code to delete all child data corresponding to the top level parent, update the top level parent, and insert the new child data. You use the ReplaceChildren element to specify what child elements Oracle Transportation Management should replace. The remaining elements are processed using the IU transaction code.
- RP: Replace Primary/Parent. Use this transaction code to replace the primary/parent object without replacing the child objects. This will null out all fields in the primary/parent object that are not contained in the incoming xml, and will perform an insert/update on all the child data.
- R: Replace. Use this transaction code to replace the primary/parent and child objects. This is a combination of the RC and RP transaction codes.

Use Cases for Transaction Codes Starting with "R"

We will use an XML file with the following Shipment XML structure to illustrate the basic rules for transaction codes starting with "R".

```
Shipment
   ShipmentHeader
          TransactionCode RC
   ShipmentStop=1
          ShipmentStopDetail: ShipUnitGid=ShipUnit A
          ShipmentStopDetail: ShipUnitGid=ShipUnit B
   ShipmentStop=2
          ShipmentStopDetail: ShipUnitGid=ShipUnit A
          ShipmentStopDetail: ShipUnitGid=ShipUnit B
   ShipUnit: ShipUnitGid=ShipUnit A
          ShipUnitContent: LineNumber=0
          ShipUnitContent: LineNumber=1
   ShipUnit: ShipUnitGid=ShipUnit B
          ShipUnitContent: LineNumber=3
          ShipUnitContent: LineNumber=4
   SEquipment: SEquipment_A
```

If the data is not found in database, the shipment is inserted into database. In this case, the RC is equivalent to IU. The use cases described in the following sections are edited from this XML.

1. Remove ShipmentStop 2 and two shipUnitContents of ShipUnit A from the above XML.
   - Expected result: ShipmentStop 2, two shipUnitContents as well as their corresponding children are deleted from database.

```
Shipment
   ShipmentHeader
          TransactionCode RC
   ShipmentStop=1
          ShipmentStopDetail: ShipUnitGid=ShipUnit A
          ShipmentStopDetail: ShipUnitGid=ShipUnit B
```



```
    <!-- ShipmentStop=2
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B -->
    ShipUnit: ShipUnitGid=ShipUnit A
           ShipUnitContent: LineNumber=0
           ShipUnitContent: LineNumber=1
    ShipUnit: ShipUnitGid=ShipUnit B
           <!--ShipUnitContent: LineNumber=3
           ShipUnitContent: LineNumber=4 -->
    SEquipment: SEquipment_A
```

2. Remove ShipmentStop 2 from the original XML and add ManagedChild=ShipmentStop.
   - Expected result: ShipmentStop 2 and all its child tables are deleted from database.

```
Shipment
    ShipmentHeader
           TransactionCode RC
           ManagedChild =ShipmentStop
    ShipmentStop=1
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B
    <!-- ShipmentStop=2
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B -->
    ShipUnit: ShipUnitGid=ShipUnit A
           ShipUnitContent: LineNumber=0
           ShipUnitContent: LineNumber=1
    ShipUnit: ShipUnitGid=ShipUnit B
           ShipUnitContent: LineNumber=3
           ShipUnitContent: LineNumber=4
    SEquipment: SEquipment_A
```

If the Is_permanent of the above stop in database equals true or the ManagedChild is set to a value other than ShipmentStop or ShipmentStopDetail, the stop as well as it child tables are not able to be removed from database.

3. Remove ShipmentStopDetail elements of ShipmentStop 2 and add ManagedChild = ShipmentStop from the original XML.
   - Expected result: ShipmentStopDs of ShipmentStop 2 are deleted from database.

```
Shipment
    ShipmentHeader
           TransactionCode RC
           ManagedChild =ShipmentStop
    ShipmentStop=1
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
           <!-- ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B -->
    ShipUnit: ShipUnitGid=ShipUnit A
           ShipUnitContent: LineNumber=0
           ShipUnitContent: LineNumber=1
    ShipUnit: ShipUnitGid=ShipUnit B
           ShipUnitContent: LineNumber=3
           ShipUnitContent: LineNumber=4
    SEquipment: SEquipment_A
```

You can set ManagedChild=ShipmentStopDetail in order to get the same result.

4. Remove ShipmentStopDetail elements of ShipmentStop 2 and ShipUnitContent elements of ShipUnit A with ManagedChild= ShipmentStop from the original XML.

   - Expected result: ShipmentStopDs for ShipmentStop 2 are deleted. SShipUnitLines are unchanged. In theory, SShipUnitLines should be deleted too. However, this is an exception case since ShipUnit, SEquipment and Text are not really child or grand child nodes of shipment. They can independently exist in database. In order to delete ShipUnitContent, you have to specify the ShipUnitContent in ManagedChild as described in next section

```
Shipment
    ShipmentHeader
           TransactionCode RC
           ManagedChild =ShipmentStop
    ShipmentStop=1
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
           <!-- ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B -->
    ShipUnit: ShipUnitGid=ShipUnit A
           <!-- ShipUnitContent: LineNumber=0
           ShipUnitContent: LineNumber=1 -->
    ShipUnit: ShipUnitGid=ShipUnit B
           ShipUnitContent: LineNumber=3
           ShipUnitContent: LineNumber=4
    SEquipment: SEquipment_A
```

5. Remove ShipUnitContent elements of ShipUnit A with ManagedChild=ShipUnit or ManagedChild=ShipUnitContent.

   - Expected result: The SShipUnitLines 0 and 1 are deleted from database. SShipUnit in the XML is replaced.

```
Shipment
    ShipmentHeader
           TransactionCode RC
           ManagedChild = ShipUnitContent
                  OR
           ManagedChild = ShipUnit
    ShipmentStop=1
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
           ShipmentStopDetail: ShipUnitGid=ShipUnit A
           ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipUnit: ShipUnitGid=ShipUnit A
           <!-- ShipUnitContent: LineNumber=0
           ShipUnitContent: LineNumber=1 -->
    ShipUnit: ShipUnitGid=ShipUnit B
           ShipUnitContent: LineNumber=3
           ShipUnitContent: LineNumber=4
    SEquipment: SEquipment_A
```

6. Add two or more ManagedChild elements (ManagedChild=ShipmentStop, ManagedChild=ShipUnit) and remove ShipmentStopDetail elements in ShipmentStop 2 and ShipUnitContent elements of ShipUnit A.

   - Expected result: ShipmentStopDs of ShipmentStop 2 and SShipUnitLines of ShipUnit A are unchanged.

```
Shipment
    ShipmentHeader
```



```
        TransactionCode RC
        ManagedChild =ShipmentStop
        ManagedChild =ShipUnit
    ShipmentStop=1
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
        <! -- ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B -->
    ShipUnit: ShipUnitGid=ShipUnit A
        <!-- ShipUnitContent: LineNumber=0
        ShipUnitContent: LineNumber=1 -->
    ShipUnit: ShipUnitGid=ShipUnit B
        ShipUnitContent: LineNumber=3
        ShipUnitContent: LineNumber=4
    SEquipment: SEquipment_A
```

7. Remove ShipUnit A and the ShipUnitContent 3 and 4 for the remaining ShipUnit B with ManagedChild=ShipUnit.
    - Expected result: SShipUnit corresponding to ShipUnit A is unchanged. SShipUnit corresponding to ShipUnit B is replaced. The SShipUnitLines corresponding to ShipUnitContents 3 and 4 are deleted from database.

```
Shipment
    ShipmentHeader
        TransactionCode RC
        ManagedChild = ShipUnit
    ShipmentStop=1
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    <!-- ShipUnit: ShipUnitGid=ShipUnit A
        ShipUnitContent: LineNumber=0
        ShipUnitContent: LineNumber=1 -->
    ShipUnit: ShipUnitGid=ShipUnit B
        <!-- ShipUnitContent: LineNumber=3
        ShipUnitContent: LineNumber=4 -->
    SEquipment: SEquipment_A
```

    - The same result can be achieved through specifying the transaction code R and ManagedChild = ShipUnitContent in ShipUnit B.

```
Shipment
    ShipmentHeader
        TransactionCode RC
    ShipmentStop=1
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    <!-- ShipUnit: ShipUnitGid=ShipUnit A
        ShipUnitContent: LineNumber=0
        ShipUnitContent: LineNumber=1-->
    ShipUnit: ShipUnitGid=ShipUnit B
        TransactionCode= R
        ManagedChild = ShipUnitContent
```



```
        <!-- ShipUnitContent: LineNumber=3
        ShipUnitContent: LineNumber=4 -->
    SEquipment: SEquipment_A
```

8. Change ManagedChild = SEquipment with TransactionCode= RC in ShipmentHeader.
   - Expected result: SEquipment_A data in database will be replaced with SEquipment data.

```
Shipment
    ShipmentHeader
        TransactionCode RC
        ManagedChild = SEquipment
    ShipmentStop=1
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipUnit: ShipUnitGid=ShipUnit A
        ShipUnitContent: LineNumber=0
        ShipUnitContent: LineNumber=1
    ShipUnit: ShipUnitGid=ShipUnit B
        ShipUnitContent: LineNumber=3
        ShipUnitContent: LineNumber=4
    SEquipment: SEquipment_A
```

9. Remove ShipmentStop 2 with ManagedChild=ShipmentStop from the original XML and change the transaction code to R.
   - Expected result: ShipmentStop 2 and all its child tables are deleted. The shipment table is replaced (This is different from transaction code RC).

```
Shipment
    ShipmentHeader
        TransactionCode RC
        ManagedChild =ShipmentStop
    ShipmentStop=1
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipmentStop=2
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    ShipUnit: ShipUnitGid=ShipUnit A
        ShipUnitContent: LineNumber=0
    SEquipment: SEquipment_A
```

10. Change the transaction code to RP in the XML above.
    - Expected result: Only shipment table is replaced. The ManagedChild element is ignored. ShipmentStop 2 and all its child tables are unchanged.

```
Shipment
    ShipmentHeader
        TransactionCode RC
        ManagedChild =ShipmentStop
    ShipmentStop=1
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B
    <!-- ShipmentStop=2
        ShipmentStopDetail: ShipUnitGid=ShipUnit A
        ShipmentStopDetail: ShipUnitGid=ShipUnit B -->
    ShipUnit: ShipUnitGid=ShipUnit A
```



```
        ShipUnitContent: LineNumber=0
    SEquipment: SEquipment_A
```

The value for element ManagedChild is defined in CHILD_ELEMENT_ALIAS column of INT_MANAGED_CHILDREN_MAP table. This value can be viewed or edited from Power Data page found by navigating to **Business Process Automation > Power Data > Integration > Managed Children Map**.

Uppercase GIDs, XIDs, and Data

It is a good idea to only use uppercase characters in GIDs and XIDs in your transmissions. This is because Oracle Transportation Management web pages only search for uppercase text strings. If you created a record with a lowercase XID using an XML transmission, it will be hard to find that record using Oracle Transportation Management web pages.

> **Note:** You should not create GIDs with trailing spaces, as these records will not be able to be looked up via the UI.

Oracle Transportation Management provides functionality to automatically change text to upper case when processing the inbound XML. You can enable the functionality by setting the following property:

```
glog.integration.enableCaseChange=true
```

If there are some elements that you do not want to change, you can set the glog.integration.casechange.element property for those elements. Valid values are element names. For example, to prohibit an argument name and value from changing, you would define the following:

```
glog.integration.casechange.element=ArgName
glog.integration.casechange.element=ArgValue
```

This is important if integrating with Oracle E-Business Suite (EBS) since data from Oracle E-Business Suite is sent as mixed case text.

Time Zone in Integration XML

Time Zone Override

Generally, time information sent in the inbound XML to Oracle Transportation Management is associated with a time zone for a particular location and time zone offset. For example, the early pickup dates and late delivery dates on a transportation order line (within the TransOrder interface) are associated with the ship from and ship to locations for the order line, and each of those locations is associated with a specific time zone. The time information sent in the early pickup date and late delivery dates elements are assumed to be in the time zone of the location. So if your time is at 8 AM in King of Prussia, PA, you would specify the time in the XML as 080000 and Oracle Transportation Management would understand this to mean 8AM in the America/New_York time zone.

For systems that only maintain their times in a single time zone, or in other circumstances when you are unable to specify times as expected in the time zone of the corresponding locations, an override can be specified to indicate the time zone for all times for the transaction. As an example, if your sending application maintains all times in a specific time zone such as San Francisco where the early pickup date is maintained as 5 AM, it would be possible to send 050000 as the time to Oracle Transportation Management and indicate an override that all the times in the transportation order are specified in the America/Los_Angeles time zone.

The time zone override can be specified in the Transmission/TransmissionBody/GLogXMLElement/TransactionHeader/TimeZoneGid element. When

the TimeZoneGid is specified, it will be assumed that all the times within the transaction (GLogXMLElement) are in that time zone. The only restriction for the value is that the value for the TimeZoneGid must correspond to one of the valid values in our time_zone table.

Addition of TimeZone information to all XML Date elements

All date/time XML elements have been updated to include the time zone and time zone offset. All outbound date elements include the new time zone information.

The original XML Date elements as strings are replaced with new elements to include GLogDate, TimeZone ID and TimeZone Offset. These new date elements are renamed with a suffix "Dt" for uniformity. Also, the original XML Date elements with GLogDate are modified to include TimeZone ID and TimeZone Offset.

For example:

Old XML Element	**New XML Element**
<StartDate>20071012173600</StartDate>	<StartDt> <GLogDate>20071012173600</GLogDate> <TZId>America/New_York</TZId> <TZOffset>-4</TZOffset> </StartDt>
< ActivityDate> <GLogDate>20071012173600</GLogDate> < /ActivityDate>	< ActivityDate> <GLogDate>20071012173600</GLogDate> <TZId>America/New_York</TZId> <TZOffset>-4</TZOffset> < /ActivityDate>

For inbound integration the old date elements are compatible and remain same, Oracle Transportation Management will convert and persist them. Backward compatibility logic is used to support inbound transmissions by enabling the property `glog.integration.enableTimeZoneCompatibility=true`.

Properties are used to specify the compatibility of old date elements to new date elements as `glog.integration.timeZone.{old element name}={new element name}`. For example: `glog.integration.timeZone.StartDate = StartDt`

XSL Transformation Files can be used for mapping elements if needed, such as `(GLogXML_v55_to_v60_DateTime.xsl, GLogXML_v60_to_v55_DateTime.xsl)`

Business Number Generator (BNG)

You can send a transmission to Oracle Transportation Management without entering a value in the GID elements in the XML transmission (for example, Ship Unit ID, Order Release ID, Order Base ID, etc.). Oracle Transportation Management generates values for these fields based on the default

business number rules when the transmission comes into Oracle Transportation Management. You can set up the BNG to create numbers that fit your needs.

Note: This only works for Transaction Code I.

Control Validation of Inbound Transmissions

After Oracle Transportation Management has processed a transmission, Oracle Transportation Management sends back a TransmissionReport to the external system with a list of validation and/or processing errors.

System administrators can set the default level of validation that Oracle Transportation Management performs. Changing the validation level can improve performance by removing unnecessary queries and logic for validating the data. The default level of validation is specified via the following "glog.integration.validation" property. Refer to the online help for the list of possible values for the property.

If your Oracle Transportation Management installation validates all transmissions fully, you can skip validation for certain transmissions on a case-by-case basis. Just include this processing instruction in all transmissions where Oracle Transportation Management should skip validation:

```
<?gc3-xml-process validate_required_fields="N"?>
```

If instead your Oracle Transportation Management installation never validates errors or only validates when receiving persist errors from the Oracle Database, you can get Oracle Transportation Management to validate certain transmissions on a case-by-case basis. Include this processing instruction in all transmissions Oracle Transportation Management should validate:

```
<?gc3-xml-process validate_required_fields="Y"?>
```

Blank Out Certain Fields

To delete (null out) values from certain fields in a record (without deleting the whole record), you can specify the (~) character in the element. For example, if a value was entered for the External System ID field in the TransOrderHeader, and that value needed to be removed in a subsequent TransOrder update, the following would be specified for the XML: <ExternalSystemId>~</ExternalSystemId>

Searching for GIDs Using Integration Saved Queries

You can select or identify information to update or delete without using a GID by using a configurable matching integration saved query. Integration Saved Queries are defined in Power Data via the following menu: **Business Process Automation -> Power Data -> Integration -> Integration Saved Queries**. The queries are written as SQL statements that contain references to the information in the incoming XML transmission. For example, a query for a shipment GID given a shipment reference number would be as follows (e.g. query "GUEST.TEST_SAVED_QUERY_001"):

```
select s.shipment_GID from shipment_refnum s where s.shipment_refnum_qual_GID =
'{%QUAL%}' and s.shipment_refnum_value = '{%VALUE%}'
```

Then your inbound XML would contain the following IntSavedQuery element to use the above query:

```
<IntSavedQuery>
   <IntSavedQueryGid>
          <Gid>
                <DomainName>GUEST</DomainName>
                <Xid>TEST_SAVED_QUERY_001</Xid>
```



```
            </Gid>
        </IntSavedQueryGid>
        <IntSavedQueryArg>
            <ArgName>QUAL</ArgName>
            <ArgValue>ZZ</ArgValue>
        </IntSavedQueryArg>
        <IntSavedQueryArg>
            <ArgName>VALUE</ArgName>
            <ArgValue>MY_SHIP_REFNUM_001</ArgValue>
        </IntSavedQueryArg>
    <IsMultiMatch>Y</IsMultiMatch>
    <NoDataFoundAction></NoDataFoundAction>
    </IntSavedQuery>
```

During processing, Oracle Transportation Management would then replace the '{%QUAL%}' with 'ZZ' and the '{%VALUE%}' with ' MY_SHIP_REFNUM_001' to search for the shipment GID. And the transaction would be processed with using the shipment GID returned from the query. Note that the IsMultiMatch element can be used to indicate if multiple GIDs can be returned from the query and used for processing. Refer to the schema notes for additional fields that are available for the query.

The queries can also be written using XPath expressions to search for values from specific elements in the XML. An example of a query that relies on XPath expressions is as follows:

```
select ob.ORDER_BASE_GID from OB_REFNUM ob where ob.ORDER_REFNUM_QUAL_GID =
'PO' and ob.OB_REFNUM_VALUE =
'{TransOrder/TransOrderHeader/OrderRefnum[OrderRefnumQualifierGID/GID/Xid='PO'
and (not(OrderRefnumQualifierGID/GID/DomainName) or
OrderRefnumQualifierGID/GID/DomainName = '' or
OrderRefnumQualifierGID/GID/DomainName = 'PUBLIC')]/OrderRefnumValue}'
```

Note that the preferred method is to use the ArgName and ArgValue as they perform much better than the XPath expressions.

If NoDataFoundAction is not null and the intSavedQuery returns no value, then transaction code for the XML is switched to the transaction code specified by NoDataFoundAction, which must be a valid transaction code.

Default Integration Saved Queries for Updates

There are a few interfaces that support default integration saved queries without having to specify the IntSavedQuery element in the inbound XML. This functionality is supported for Location, TransOrder, TransOrderLine, and Shipment.

The default integration saved queries are only used when the transaction code element has a value of U or D, and the primary GID for the interface is missing. If a TransOrderGid is missing, then the INT_TRANS_ORDER_GID_1 and INT_TRANS_ORDER_GID_2 saved queries are used. If a TransOrderLineGid is missing, then the INT_TRANS_ORDER_LINE_GID_1 and INT_TRANS_ORDER_LINE_GID_2 saved queries are used. If a Shipment GID is missing, then the INT_SHIPMENT_GID_1 and INT_SHIPMENT_GID_2 saved queries are used. If a Location GID is missing, then the INT_LOCATION_GID_1 and INT_LOCATION_GID_2 saved queries are used.

The defined queries must return a single GID of the element being referenced (for example, Order_Base_GID for TransOrder, Shipment_GID for Shipment, etc.). When a query returns multiple records, it will generate an error. Oracle Transportation Management supports up to two queries for each interface/record. If the first query generates an error or does not resolve to a single record, the second query will be applied. And if nothing is returned, then an error is generated.

Including Non 7-Bit ASCII Characters

To be able to send transmissions to Oracle Transportation Management containing characters outside the 7-bit ASCII character set, you must:

- Make sure your database uses an encoding that can handle all the characters you need.
- Specify that same encoding in your XML file. For example, <?xml version="1.0" encoding="UTF-8"?>

The default character encoding for inbound XML transmissions is UTF-8. Oracle Transportation Management will receive transmissions in other formats as well. To accept character encoding in ISO-8859-1 format, specify it in the XML file. For example, <?xml version="1.0" encoding="ISO-8859-1"?>

- Save your XML file using that same encoding. For example in UTF-8 format.
- If sending via HTTP post, you will also need to specify the encoding in the HTTP header. You must use the "Content-Type" attribute on the HTTP header to indicate that it's a stream of character data with a given encoding.

XML Spy, Textpad, and Notepad (Microsoft Windows 2000 or better) can all save in UTF-8 format. If you use a third party integration application to send your transmission via HTTP Post, refer to their documentation for how to send XML in UTF-8 format.

Inbound Transmission Staging and Processing

Functionality has been added to support inbound staging of transmission, where transmissions can be loaded in Oracle Transportation Management and processing can be delayed until some later time or event. Transmissions are flagged for staging by setting the Transmission/TransmissionHeader/TransmissionType = " STAGING". For Transmissions that are marked as staging, the transactions can be moved/added to existing staging transmission using the TransmissionHeader/StagingInfo/StagingQuery/IntSavedQuery element: the first transmission number returned from the integration saved query that is run is used as the transmission number to append the transactions.

Also, there is a StagingInfo/StagingProcess/IntSavedQuery element which can be used to control whether the Transmission should process after it is staged. If the query returns a result, the Transmission will proceed with processing, and if no result is returned then the transmission will remain in a STAGING status until another event triggers it to be processed. Initiating the processing of the Transmission can also be enabled through the "TRANSMISSION" agent type, and the Staging Transmission Processing process management actions. Refer to the online help for details on the agents and process management actions.

Oracle Transportation Management Internal Processing

To understand how to automate integration and use agents, it helps if you know more about what Oracle Transportation Management does internally. These sections list the main internal events after you have sent a transmission or message to http://hostname/GC3/glog.integration.servlet.WMServlet.

Oracle Transportation Management Transmission Processing

If the servlet receives a Transmission XML, the integration module does the following:

1. Extracts authentication information from the TransmissionHeader. This information can be in the HTTP header instead.
2. Validates username and password.
3. Splits transmission into transactions.

4. Sends a TransmissionAck back to the sender as the synchronous response.
5. Validates the contents of the transactions. The integration module validates the following:
   - Foreign keys, for example a LocationGID must exist in the Location table.
   - Data Types, for example a number only contains numeric characters.
   - TransactionCode. If set to I, the integration module checks that the primary key does not exist. If set to U or D, the integration module checks that the primary key does exist.
   - Required Elements are not NULL.
6. Saves any validation errors.
7. If there are agents listening for pre-persist events, those agents kick in now. An example of this is the public Order Base - Insert agent for TransOrder transmissions.
8. Persist transactions to corresponding database tables. Converts strings to data types; string to Boolean, string to date, and so on.
9. If there are agents listening for post-persist events, those agents kick in now. There are a number of public agents that start at this point.
10. Sends a TransmissionReport with validation and processing errors; Depending on your property settings, Oracle Transportation Management might only send a TransmissionReport if there are errors.

Note: The Transmission Report is only physically transmitted when the all workflow that is considered to be part of the Transmission Process has completed (known as "child" processes). If there is a significant amount of agent workflow configured this could appear as a delay in delivering the report. If this is an issue it may be worth considering the use of custom events with the "Create New Process" option selected.

Oracle Transportation Management Transmission Status

Each Transmission sent into Oracle Transportation Management has a status field that indicates the state of the Transmission. You can view the status for a Transmission in the Transmission Manager UI that can be accessed via the following menu: **Business Process Automation > Integration > Transmission Manager**. The status of the transmission could be one of the following:

Status	Description
STAGED	Indicates that the Transmission is initially stored in the transmission tables, and is awaiting some pre-preprocessing steps such as sequencing of the Transactions for processing.
STAGING	Indicates that the Transmission has been staged and is waiting on a condition to be released for processing. In this status, additional transactions can be added to it prior to processing. Refer to the **Inbound Transmission Staging and Processing** section for additional details.
FRESH	Indicates that the Transmission is waiting to complete processing. The individual transactions may still be processing.
ERROR	Indicates that the Transmission had completed processing and there were errors in the processing.
PROCESSED	Indicates that the Transmission successfully completed processing.

Status	Description
REDO	Indicates that the Transmission is waiting for the REDO logic to initiate re-processing of it. Please refer to the online help for REDO processing.
MESSAGE	Indicates that the Transmission is a Message. The Message Hub Manager should be used to manage the message.

Oracle Transportation Management Message Processing

If the servlet receives a Message XML, the integration module does the following:

1. Extracts authentication information from the MessageHeader. This information can be in the HTTP header instead.
2. Validates username and password.
3. Stores an entry in the I_message table, and the content in the I_transmission table.
4. Sends a MessageAck back to the sender as the synchronous response.
5. Checks for presence of MessageTypeGid and StylesheetProfileGid. If neither are present, checks for the MessageProfileGid, and uses the StylesheetProfileGids specified in the Message Profile to determine the correct Stylesheet Profile.
6. Using the Stylesheet Profile, the message is parsed to extract designated fields and updates the message table with extracted fields as needed.
7. Using the Message Type specified, the module performs the following:
    a. Notifies Message Center recipients specified on the Message Type
    b. Associates or disassociates devices to drivers, equipments, and power units as indicated on the Message Type
    c. Generates a Transmission XML if indicated on the Message Type. The Transmission XML would then be processed as indicate in the Transmission Processing section above.
8. Sends a MessageReport with validation and processing errors; Depending on your property settings, Oracle Transportation Management might only send a MessageReport if there are errors.

Oracle Transportation Management Message Status

Each message sent into Oracle Transportation Management has a status field that indicates the state of the message. In addition, the content of the message is stored in the transmission table which also has its associated status. You can view the status for a message in the Message Hub Manager UI that can be accessed via the following menu: **Business Process Automation > Integration > Message Hub Manager**. The status of the message, and its related transmission, could be one of the following:

Message Status	Related Transmission Status	Description
RECEIVED	MESSAGE	Indicates that the message is initially received in the message table, with the content stored in the transmission table.

Message Status	Related Transmission Status	Description
PROCESSING	MESSAGE or (dependent on Transmission processing)	Indicates the message is being processed. If the Message Type for the received message indicates a Transmission is to be generated and processed, then the status of the Transmission will be initially changed to FRESH and will be handled by normal transmission processing.
PROCESSED	MESSAGE or (dependent on Transmission processing)	Indicates that the message successfully completed processing.
ERROR	MESSAGE or (dependent on Transmission processing)	Indicates that the message had completed processing and there were errors in the processing.

4. Send Data Outbound

There are various ways to send integration transmissions from Oracle Transportation Management:

- Some messages are sent automatically as the result of workflow notification.
- Send and schedule integration transmissions in the Process Manager.
- Re-send and schedule integration transmissions in the Process Manager.
- Send transportation records to external systems from various managers using the Go drop-down action. The type of information you can transmit is determined by the location in Oracle Transportation Management from which you are sending it.

  Note: Before you can send integration data to an external system, you must define the system in the External System Manager.

  Note: The default character encoding for outbound XML transmissions is UTF-8.

External User Credentials

When sending messages outbound from Oracle Transportation Management it may be required to specify the username and password required by the receiving external server. This can be achieved by using the username related fields on the external system record configured to hold the connectivity details for the external server.

The following options are available:

- Use Credential: Oracle Transportation Management username associated with the process[2] sending the message will be present in Transmission and/or Message XML header. No password will be sent.
- Username and Password fields: Appear in Transmission and/or Message XML header and also be used for transport specific user credentials (e.g. HTTP header Basic Authentication or FTP user/password). Password will be sent.
- Send Encrypted Password: If a password is sent it will be sent as a base64 encoded SHA-1 digest
- Web Service Endpoint username and password: Can be used in conjunction with Username and Password fields on External System header. The External System fields will appear in the Transmission and/or Message XML header elements whereas the Web Service Endpoint username and password will be used for client authentication (and possibly also WS-Security header. See section 11, Web Service Security (WS-Security))

Outbound XML Profiles

Out XML Profiles are used to exclude portions of outbound XML with a high degree of control. They reduce the size of the XML and minimize the number of queries that are used to generate the xml, thereby reducing the memory and time used and improving overall performance. There are several options for specify the elements to exclude. Please refer to the online help for additional details.

[2] The workflow process determines the user and will either be the user signed in to Oracle Transportation Management whose action triggers the message or the automation agent configuration ("Run As").

Transform Outbound XML with XSL

Do the following to have Oracle Transportation Management transform your outgoing transmission from GLogXML to some other XML schema.

1. Define an XSL file that transforms between the schemas.
2. Upload the XSL file.
3. Define that the XSL file should be used for a specific external system.

Support for Responses to Outbound Messages.

Functionality has been added to outbound integration processing to support receiving the TransmissionAck, TransactionAck, and TransmissionReport. You will be able to use these XML documents as a functional acknowledgement to receiving the Transmission and/or Transaction. Agent objects have been added for types "TRANSMISSION OUT" and "TRANSACTION OUT" to provide control on responding and handling of the response messaged.

Oracle Transportation Management Internal Processing

To understand how to automate integration and use agents, it helps if you know more about what Oracle Transportation Management does internally. This section lists the main internal events when you send a transmission from Oracle Transportation Management.

Oracle Transportation Management does the following:

1. Populates the internal Java object classes based on SQL Queries. Some Out XML Profiles are applied during the generation of the objects.
2. Convert the Java objects into an XML String.
3. Apply XSL Transformations if needed.
4. Save the XML (Or Transformed output) to the I_TRANSMISSION table. If the outbound integration is designated as a message (e.g. generated via the Compose and Send UI action or dispatch actions), then an entry is stored in the I_MESSAGE table.
5. Send the XML via the indicated notify type (e.g. HTTP POST). Oracle Transportation Management does not require an acknowledgment from the external system.
6. Updates the status of the Transmission.

Oracle Transportation Management Message Status

The status of the message for outbound

Message Status	Related Transmission Status	Description
PROCESSING	MESSAGE or (dependent on Transmission processing)	Indicates the message is being generated or being sent.

Message Status	Related Transmission Status	Description
PENDING	MESSAGE	Indicates the communication method is "BY DEVICE" and there is no associated device to the contact. The status will remain in PENDING until a device is associated to the recipient.
PROCESSED	PROCESSED	Indicates that the message successfully completed processing.
ERROR	ERROR	Indicates that the message had completed processing and there were errors in the processing.

Integration Unit of Measure Preferences

You can specify the Unit of Measure (UOM) preference and precision to be displayed in outbound XML by defining preference for each UOM. These preferences can be defined with the Integration Preference UI that can be accessed via the following menu: **Business Process Automation > Power Data > Integration > Integration Preferences**. Integration preferences can be associated to an external system or an out XML profile. Refer to the online help for additional details.

Integration preferences can also be associated with the Rate Inquiry and Generic Query interfaces. To use the integration preference in those synchronous interfaces, you would specify the IntPreferenceGid in the RIQQuery, or the GenericQuery XML. When specified in the RIQQuery, the RIQQueryResponse would apply the preferences before responding with the result XML. When specified in the GenericQuery XML, the preference would be applied to the resulting generated interface XML.

Excluding Namespace Names in Outbound Interfaces

Namespace names can be disabled from showing up in the outbound XML document by setting the following properties to false:

glog.integration.enableXmlNamespace = false

glog.integration.enableTargetNamespace = false

These properties will affect ALL outbound interfaces to ALL External Systems.

Note: For XML documents sent inbound to Oracle Transportation Management where the namespace may be specified for several elements (not only in the parent Transmission element), or a namespace attribute specified for each element, it may be necessary to enable the parsers in the web server (refer to Section: **Enable Parsing in Oracle Transportation Management Servlets for Namespaces**).

In addition to the above, each External System can be configured separately to control whether namespace names are to be present in the outbound message. This is achieved by the **Include Namespace** field on the External System Manager UI.

The field has three possible values:-

DEFAULT	This is the default value and corresponds to using whatever is defined on the system as whole as regards the properties mentioned above.
NONE	This will remove all namespace names from the XML
STRICT	This will ensure the namespace names are consistent with the GLogXML and GLogXML-GTM schema definition target namespaces.

5. Types of Interfaces

Oracle Transportation Management offers the following integration interfaces:

Data Loading

Interface	Description	Inbound	Outbound
ActivityTimeDef	Specifies fields for allowing multiple, fixed, and variable stop times for each location role based on transport handling unit and commodity.	In	Out
Contact	Defines a person that can be contacted through Oracle Transportation Management.	In	Out
ContactGroup	Represents a list of contacts used for notification	In	Out
CSVDataLoad	CSVDataLoad provides the capability to embed the contents of a CSV file for insertion into the database. Each CSVDataLoad element can contain only one CSV File. This element should only be used for small sets of data.	In	-
CSVFileContent	Sends and receives data in CSV format. You can use it to send any type of integration data but the most important function is to send rate offering information to Oracle Transportation Management.	In	-
Driver	Sends and receives driver data to Oracle Transportation Management which includes basic personal information, CDL information, driver types, driver team data, status, remarks, etc.	In	Out
ExchangeRate	Sends the exchange rate for a particular interface to Oracle Transportation Management.	In	Out
HazmatGeneric	Transmits hazardous material based on the shipping name for an item.	In	-
HazmatItem	Sends and receive records for particular hazardous items.	In	-
ItemMaster	Transmits item master data to Oracle Transportation Management. Item master data includes item numbers, descriptions, and packaging details. Item master data must exist in Oracle Transportation Management before you can import transportation orders.	In	Out
Itinerary	Define the path between two locations and specifies the constraints for building the shipment.	In	-
Location	A place where transportation related activities, such as loading and unloading freight, occur. In addition, a location is a corporation, and/or a service provider. Use the location element to transmit location information, for the Transportation Orders interface.	In	Out

Interface	Description	Inbound	Outbound
Mileage	Defines the distance between points for a particular lane.	In	-
PowerUnit	Sends and receives power unit (such as a tractor) data to Oracle Transportation Management. This information includes power unit name, power unit type, special services, remarks, etc.	In	Out
Equipment	Send and receives the equipment data assigned to a shipment which includes basic equipment information, equipment seal, special services, remark, reference number, etc.	In	Out
ServiceTime	Transmits the time it takes to go between the two locations of an X Lane. Service time can be included as part of the Mileage element or transmitted to Oracle Transportation Management as a stand-alone XML Element.	In	-
User	Represents an Oracle Transportation Management user.	In	-
XLane	Defines a link from a source to a destination. The source and destination may specify either general or specific geography. For example, a source could be an exact location, or an entire state.	In	-

Transport Flow

Interface	Description	Inbound	Outbound
TransOrder	Oracle Transportation Management receives transportation orders from external systems. These orders can include basic information such as IDs, pick-up and delivery dates, service providers, and details such as ship units or line items.	In	Out
OBShipUnit	Contains information on ship units in an order base.	-	Out
OBLine	Contains information on lines in an order base.	-	Out
OrderMovement	Represents part of the routing for an order. It is also a collection of ship units that are unplanned for certain part of a route.	In	
TransOrderStatus	Sets order base status events.	In	-
BulkPlan	Describes the orders that Oracle Transportation Management planned and the shipments that Oracle Transportation Management created.	-	Out

Interface	Description	Inbound	Outbound
BulkRating	Describes rating statistics on the orders that Oracle Transportation Management planned and the shipments that Oracle Transportation Management created.	-	Out
BulkTrailerBuild	Describes the shipment groups created during the bulk trailer build process.	-	Out
BulkContMove	Describes the shipments that were selected and linked during a given run of bulk continuous move.	-	Out
Release	Transmits order release information to and from Oracle Transportation Management.	In	Out
ReleaseInstruction	Allows you to specify line items and ship units, and release specific quantities of them for a particular order base.	In	-
ActualShipment	Third parties send actual shipments to Oracle Transportation Management, which define the final form of a shipment.	In	-
TenderOffer	Once a shipment has been planned, Oracle Transportation Management sends a tender offer to a service provider. The tender offer provides a contract for the service provider to carry a particular shipment.	-	Out
TenderResponse	Receives responses from service providers regarding tender offers.	In	-
ShipmentStatus	Service providers and other third parties transmit shipment event information to Oracle Transportation Management.	In	Out
ShipmentLink	Identifies related shipments at a consolidation or de-consolidation pool for both inbound and outbound interfaces.	In	-
ShipmentGroup	Specifies shipment group header information and the associated shipments in a group.	In	Out
ShipmentGroupTenderOffer	Notifies a service provider of a shipment group pickup.	-	Out
PlannedShipment	Oracle Transportation Management builds planned shipments and sends them to service providers as part of the tender process or to a warehouse management system (WMS).	-	Out

Interface	Description	Inbound	Outbound
SShipUnit	Queries for and updates a shipment/shipunit without information on what shipment(s) it belongs to.	In	-
OrderMoveReplace	Order Movement Replace (OMR) is used to bring in production lot and delivery lines information to carry out the necessary modifications to order movements.	In	-
ShipStop	Used to modify stop related information for a shipment.	In	-
CharterVoyage	Represents an ocean transport movement by a carrier from a loading port to a discharge port.	In	Out
Consol	Used to specify the shipment consolidator. A consol can be created for a charter voyage or air schedule (flight).	In	Out
BookingLineAmendment	Used to send booking line changes out of the system.	-	Out
Claim	Used to specify information for damaged shipments, can be used to notify parties of their involvement with a claim, and can tracks status changes that occur throughout the claim process	In	Out
TransOrderLink	Used to establish a link between order base objects.	In	-
DemurrageTransaction	Contains all the details related to a demurrage transaction.		Out

Financial

Interface	Description	Inbound	Outbound
Invoice	Represents what is owed to the service provider for transporting the shipment.	In	-
Billing	Sends transmissions to an accounting system for customer billing. The billing information represents the amount owed by a customer to the planner of a shipment. The billing transmission includes the customer who is being charged and details of the bill such as the amount due, the date due, and any discount information.	-	Out

Interface	Description	Inbound	Outbound
Voucher	Transmits payment information. A voucher represents the cost of a shipment owed to a third party such as a service provider.	-	Out
AllocationBase	Sends order release allocation cost information to an order owner.	-	Out
FinancialSystemFeed	Sends billing and shipment cost allocation information to an external financial system.	-	Out
WorkInvoice	Includes a record of driver activities for a shipment. This is the record sent to payroll and used to calculate driver pay.	In	Out
Accrual	Used to transmit the allocated freight cost accrued or paid against orders. This can be used to communicate changes or differences in an order's allocated freight cost to other external systems.	-	Out

Miscellaneous

Interface	Description	Inbound	Outbound
TransmissionAck	Immediately upon receiving a transmission, Oracle Transportation Management sends a receipt back to the sending system that sent the original Transmission. Also, Oracle Transportation Management can receive an inbound TransmissionAck to indicate receipt of the outbound Transmission.	In	Out
TransmissionReport	Once a transmission is processed, Oracle Transportation Management sends a report that indicates any problems with the transmission. Also, a TransmissionReport can be sent inbound to acknowledge receipt and processing of a Transmission.	In	Out
TransactionAck	Acknowledges receipt and processing of a Transaction.	In	-
RemoteQuery	Queries Oracle Transportation Management for rate information, based on quantity, locations, and/or dates of a shipment.	In	-
RemoteQueryReply	Provides Oracle Transportation Management's response to a remote query about rates and shipments sent by a customer. Remote queries are used to gather information (for example, a customer can ask for rates based on quantity, locations, and dates).	-	Out

Interface	Description	Inbound	Outbound
DataQuerySummary	Sends a summary of the data required by an external system.	-	Out
Job	Sends data that addresses how logistics services providers and freight forwarders manage the services they provide within Oracle Transportation Management. A Job offers a workspace that brings together the objects and activities required of them.	In	Out
RATE_OFFERING	Is a general contract with a service provider. It indicates what rate offering data was used to rate the shipment.	-	Out
RATE_GEO	Provides specific costing or rating data from one place to another. It indicates what rate record data was used to rate the shipment.	-	Out
Schedule	Sends schedules as input to the processes for building shipments and assigning orders into batches.	In	Out
Sku	Defines a stock keeping unit including what quantities to keep in stock, and the actual amount in the warehouse.	In	-
SkuEvent	Specifies a SKU event, which describes activities on SKU's	In	-
SkuTransaction	Represents a shipment of SKUs arriving or leaving the warehouse.	In	-
Topic	Raises a topic and gets Oracle Transportation Management to start processing an object.	In	-
GenericStatusUpdate	Updates the external statuses of Locations, TransOrders, Payments, OrderReleases, Shipments, Vouchers, ShipmentGroups, and Schedules.	In	-
RouteTemplate	Route template represents the plan for a cooperative route. A cooperative route is a linking of lanes that have been identified to have sufficient recurring volume of shipments to form a good route for a fleet or dedicated vehicle.	In	Out
Voyage	Used to send world-wide vessel schedule information to external systems.	-	Out
Document	Provides a consistent way to send and receive business documents in and out of the system	In	Out
DriverCalendarEvent	Specifies a driver calendar event, which describes calendar event information on a driver.	In	

Interface	Description	Inbound	Outbound
Quote	The Quote Interface allows customer service representatives to supply their customers with transportation quotes.	-	Out
DemurrageTransaction	The Demurrage Management Solution deals with business aspects related to demurrage which is a term for the fees charged by a carrier to a customer for the use of assets beyond a contracted free time for loading and unloading	-	Out

Global Trade Management Interfaces

Interface	Description	Inbound	Outbound
ServiceRequest	ServiceRequest is used to invoke a screening service in the Global Trade Management product. Such services include restricted party, santioned party, classification, and compliance rule.	In	
ServiceResponse	Provides the response to the ServiceRequest.		Out
GTMItem	Represents a Global Trade Management item/product.	In	Out
GtmParty	Represents a Global Trade Management party.	In	Out
GtmTransaction	Represents a Global Trade Management trade transaction.	In	Out
GtmRegistration	Represents a Global Trade Management registration.	In	Out
GtmShipment	Represents a Global Trade Management Customs Shipment	In	Out

6. Setting Up Interfaces

The following general information helps you set up your interfaces. If an interface has specific setup requirements, they are found with the pages defining each interface.

To set up interfaces, you must define where to send transmissions and what to do with the transmissions Oracle Transportation Management receives. Information throughout Oracle Transportation Management acts interdependently; one piece of information depends on another to perform an action. For some interfaces to work, data from other sources must already be present in Oracle Transportation Management. For example, before you can create a shipment, you must create itineraries.

Define External Systems

To send transmissions to other systems, you must define the systems in Oracle Transportation Management using the External System Manager.

User Management

You must add service providers as users and enter user associations for them. To perform user management functions, log in to the SERVPROV with a username that contains administrator (ADMIN) rights.

- Define service providers as users in Oracle Transportation Management.
- Define associations for the service providers.

Workflow Parameters

In Power Data, define workflow parameters that determine how Oracle Transportation Management responds to inbound and outbound transmissions. You define Workflow Power Data topics to define the way the tendering shipments works.

- Workflow Parameters: Use the Workflow Parameters to define how Oracle Transportation Management tenders shipments. You also define shipment notification messages. For example, you define information, warning, and, fatal messages that Oracle Transportation Management sends out as the results of status information sent by service providers about particular shipments.
- Workflow Trigger Parameters: Use the Workflow Trigger Parameters to define how often Oracle Transportation Management performs tender activity. This topic helps you control system performance. For example, if Oracle Transportation Management is performing tender actions too frequently, your system performance may be slowed.

Agent Manager

The Automation Agent Manager lets you construct workflow agents that are key components to automate Oracle Transportation Management. A workflow agent listens for an Oracle Transportation Management event, verifies a user-defined condition, and executes one or more actions that you choose from an action library.

7. Interfaces

You can view diagrams of the XML schema for the interfaces by viewing the XSD file with an XML management tool.

ProcessInfo

ProcessInfo controls how Oracle Transportation Management processes GLogXMLElement elements. You can think of ProcessInfo as a transaction header.

In the WhenToProcess element, you can specify "END_OF_TRANSMISSION" or leave it unspecified. If you specify END_OF_TRANSMISSION, then Oracle Transportation Management processes that transaction after all other transactions in the transmission. This setting is useful for making sure that Oracle Transportation Management processes a Topic transaction last in a transmission.

If you leave WhenToProcess unspecified, then Oracle Transportation Management processes as normal and according to the other ProcessInfo elements.

> **Note:** Oracle Transportation Management ignores the WhenToProcess element if you set IsProcessInSequence=Y in the TransmissionHeader.

Shipment Interfaces (INS)

Shipment interfaces work for both outbound and inbound processing. For example, you can send shipment transmissions to service providers as part of the tender process and receive actual shipment transmissions back from them representing what is actually being shipped. In addition, shipments can be sent to Oracle Transportation Management for processing that do not have order information associated with them.

> **Note:** All shipments use the same interface schema diagrams.

Planned Shipments

Use this interface to send planned shipments from Oracle Transportation Management to an external system. Planned shipments perform two functions:

If a warehouse management system (WMS) is defined as one of the involved parties for the shipment, Oracle Transportation Management automatically sends a transmission containing the shipment information to the WMS to determine if the items that are being shipped are available. For example, Oracle Transportation Management sends a planned shipment to a warehouse, requesting 5000 pounds of food be shipped to a customer.

Oracle Transportation Management sends the planned shipment as part of a tender offer to the service provider associated with the shipment. Planned shipment transmissions get sent to service providers automatically when a tender offer is made or withdrawn.

Actual Shipments

Transmissions in the actual shipment interface define working shipments that are sent to Oracle Transportation Management by third parties, such as service providers. For example, when a shipment is tendered to a service provider, a copy of the planned shipment is included with the tender offer. The service provider responds to the tender and, if the service provider accepts, sends the actual shipment transmission back to Oracle Transportation Management. The actual shipment represents the working shipment that is being transported (A planned shipment represents what Oracle Transportation Management expects the shipment to contain).

If the service provider sends a new order release as part of the actual shipment, Oracle Transportation Management creates an order release and order base for the new release.

An actual shipment is required to print shipment documentation such as a bill of lading or Domestic Packing List.

Actual Shipment Workflow Considerations

Changes to an existing shipment using the Actual Shipment interface can be configured to trigger additional business logic by way of agent workflow. When the shipment contains multiple orders or is one of a 'graph' of multiple shipments for different modes or service providers, this workflow can be quite complex and involve modification of other objects. Consequently, although simple modifications **CAN** be achieved using the Actual Shipment interface, e.g. addition of a new Shipment Reference Number, these cases should instead make use of the Generic Status Update interface (see section GenericStatusUpdate Interface).

Structural Changes to Shipment Ship Units in Release 6.0

Prior to Oracle Transportation Management release 6.0, shipment ship units were shared among different shipments/legs of a multi-leg movement. With the multi-tier execution enhancement in release 6.0, this has been changed to not allow sharing of the shipment ship units across multiple shipments. As a result of this change, updates to Shipment Ship Unit details (quantities or attributes) will not automatically be reflected across all legs when a change is made to a shipment ship unit. There are several options for making the appropriate changes to each of the impacted legs:

- Update Each Shipment Manually
    - Update each of the shipments impacted with a separate ActualShipment xml. Each ActualShipment xml would update the appropriate ship units on that shipment.
- Leverage Integration Saved Query
    - Leverage the IntSavedQuery within the Shipment/ShipmentHeader and Shipment/SShipUnit to search for the GIDs of the shipment(s) and ship unit(s).
    - This option is beneficial when the shipment and/or ship unit GIDs are not all known in the integration layer.
    - This option is limited in that any other fields set in the ActualShipment would be applied to all shipments identified by the query. A similar situation exists for the Ship Units on the shipments.
- Use Propagation for Ship Unit Changes
    - This is useful to propagate the count and quantity changes to upstream or downstream shipments.
    - Initiated by setting the `Shipment/ShipmentHeader/IntCommand/IntCommandName = "PropagateShipUnitChanges"`. The propagation option (upstream, downstream, or both) is specified by the IntCommand/IntArg element. Refer to the GLog XML Schema for the specific values to be used.
    - This option has limitations on the types of changes it will handle and propagate. Several are listed as follows:
        - Cannot Add SEquipment From Another Shipment: You cannot add a SEquipment that already exists on another shipment to a shipment with the propagation. The SEquipment addition should be done in a separate ActualShipment without the propagation.
        - Cannot Change SEquipment for Existing Ship Unit: If there are more than one SEquipment on a Shipment, you cannot change the SEquipment for an existing Ship Unit with the propagation.

- Cannot Add New SEquipment for Existing Ship Unit: You cannot add a new SEquipment and assign to an existing Ship Unit with the propagation. The SEquipment changes should be done in a separate ActualShipment without the propagation.
- Limited Support to Add New SEquipment and New Ship Unit: You can add a new SEquipment and assign to a new Ship Unit. The logic will determine the new SEquipment and persist it before initiating the propagation logic. The ability to support adding the new SEquipment is controlled by the following property: glog.integration.shipment.persistNewSEquipmentForShipmentActualAPI=true
- Can Add a New Ship Unit: The propagation supports adding a new Ship Unit to the Shipment. The Ship Unit must be assigned to an existing SEquipment on the shipment.
- Cannot Use DR Transaction Code In Ship Unit: You cannot use the DR transaction code in the Ship Unit to dereference the Ship Unit from the Shipment. Since the Ship Units are no longer shared among shipments, you should consider using the D transaction code to delete the ship unit. If there is a need to use the DR transaction code, it should be done in a separate ActualShipment without the propagation.

Note: When migrating to Oracle Transportation Management 6.0 from a previous release, review the implementation of the Actual Shipment integration into Oracle Transportation Management to determine the impacts of this change.

Updating Parts of a Shipment

When sending an actual shipment to Oracle Transportation Management you often want to update parts of an existing shipment. Generally the TransactionCode of the shipment (ShipmentHeader/TransactionCode) provides the guiding rule for the child elements. Here are some examples:

Element	Description
Shipment/ShipUnit/ShipUnitContent	If TransactionCode=IU, a LineNumber that does not exist will be added, otherwise updated. Currently, you cannot delete an individual line.
Shipment/ShipmentHeader/ShipmentRefnum	If TransactionCode=IU, a new QualifierValue pair will be added. You can delete (and replace) using the GenericStatusUpdate interface.

Element	Description
ShipmentHeader/Remark	If TransactionCode is IU and the RemarkSequence does not exist, then Oracle Transportation Management will automatically generate a sequence number and add the remark. If TransactionCode is IU and the RemarkSequence does exist, Oracle Transportation Management updates with a new RemarkQualifier and RemarkText. If you supply neither a RemarkSequence nor a RemarkQualifier, Oracle Transportation Management adds the RemarkText as new Remark. You can delete (and replace) using the GenericStatusUpdate interface.

In the Shipment element, if you set the transaction code to RC and set the ReplaceChildren/ManagedChild element to "ShipmentStop", Oracle Transportation Management deletes all shipment stops for that shipment and replaces the deleted shipment stops with the shipment stops from your transmission.

> **Note:** This does not apply to shipment stops marked IsPermanent (same as Permanent check box in Oracle Transportation Management web interface).

In the Shipment element, if you set the transaction code to RC and set the ReplaceChildren/ManagedChild element to ShipmentStopDetail, Oracle Transportation Management replaces the existing ship units with the ship units in your transmission.

> **Note:** This does not apply to existing ShipmentStopDetails marked IsPermanent.

> **Note:** Within ShipmentStopDetail, the removal of the reference to the ShipUnitGID(s) will not remove the S_Ship_Unit from the system. Only the reference to the object is removed.

In the Shipment element, if you set the transaction code to U and the Shipment ID is missing from either the transaction or the database, you will receive an error.

When a new shipment referencing a ship unit is added with missing ship unit data, then the ship unit data is pulled from the database. When a new shipment referencing a ship unit is added with new data, then the ship unit data passed in through integration is used.

Adding Stops

There is no way to insert a new stop to a shipment via shipment actuals unless the new stop has a stop number that does not already exist on the shipment (like adding stop #3 to a 2-stop shipment, or adding stop #2 to a shipment with stops 1 and 99).

Adding Ship Units

An added ship unit should be linked to an order release that is on a shipment (this order release must be planned on the initial shipment) and should be linked to the initial pickup location. If the flag on the Shipment Header indicates “Propagate Updates,” the Oracle Transportation Management integration layer will call business logic to add the new ship unit to subsequent stop on the initial shipment and all affected succeeding shipments.

To add a ship unit to a shipment, the following must be done in the ActualShipment XML interface:

1. Specify a flag to indicate that new ship unit should be applied to downstream shipments.

   ```
   AcutalShipment.Shipment.ShipmentHeader.IntCommand.IntCommandName =
   "PropagateShipUnitChanges"
   ```

2. Indicate that DropOff stop should be determined for the shipment, but not to propagate the ship unit changes.

   ```
   ActualShipment.ShipmentHeader.IntCommand.IntCommandName =
   "DetermineShipUnitDropoff"
   ```

3. Specify the new ship unit.

   ```
   ActualShipment.ShipmentHeader.Shipment.ShipUnit.ShipUnitGID =
   ```

4. Specify the Transaction Code (optional).

   ```
   ActualShipment.ShipmentHeader.Shipment.ShipUnit.TransactionCode = "I" or
   "IU"
   ```

5. Specify the pickup stop for the ship unit.

   ```
   ActualShipment.ShipmentHeader.Shipment.ShipmentStop.ShipmentStopDetail.Activ
   ity = "P"
   ActualShipment.ShipmentHeader.Shipment.ShipmentStop.ShipmentStopDetail.ShipU
   nitGID = ShipUnit.ShipUnitGID
   ```

6. ShipmentFrom and ShipmentTo Locations can in the Shipment.ShipUnit element are ignored. They will be based on the release.
7. Assign an SEquipment for the new ShipUnit via one of the following options:
8. Using the ShipUnit.SequipmentGID element
9. Allow integration to assign it by using ActualShipment.ShipmentHeader.Shipment.ShipUnit.SEquipmentGIDQuery.SequipGIDMatchOption = "Any"
10. Query the SEquipmentGID using the ActualShipment.ShipmentHeader.Shipment.ShipUnit.SEquipmentGIDQuery.IntSavedQuery
11. Have the business logic assign it by not specifying the element in the ship unit.

Updating Ship Units

The updating of ship units means packaged items can be deleted or added to those ship units. Quantities from existing items can also be changed.

The following options are available via integration:

In the SShipUnit XML interface:

```
SShipUnit.TransactionCode = "RC"
SShipUnit.ReplaceChildren.ManagedChild = "ShipUnitContent"
```

In the ActualShipment XML interface:

```
Shipment.ShipmentHeader.TransactionCode = "RC"
Shipment.ShipmentHeader.ReplaceChildren.ManagedChild = "ShipUnitContent"
```

In the Shipment.ShipUnit.SShipUnit XML interface:

```
ActualShipment.Shipment.ShipUnit or ActualShipment.Shipment.SShipUnit
SShipUnit adds the ability to query for the ShipUnitGID if it's not known
```


Deleting Ship Units

Deleting a ship unit only removes the link between the shipment stop and the ship unit, as well as the link between the ship unit and equipment. The actual ship unit will not be deleted from the database. integration will also attach the ship unit remark, "Ship Unit Not Picked Up" to the ship unit.

A ship unit can be marked for removal from the shipment via the TransactionCode as follows:

```
ActualShipment.Shipment.ShipUnit.TransactionCode = "DR"
```

where "DR" corresponds to "Delete Reference." The ship unit will be removed from the shipment, but not deleted from Oracle Transportation Management.

Alternatively, you can delete ship units from a shipment using the IntCommand via integration. You can either delete all the ship units from the shipment, or only those that are marked as non-permanent. When used, the ship unit record, its shipment stop detail record, and any corresponding equipment, is deleted.

Specify the integration command as follows:

To remove all ship units:

```
<IntCommand>
   <IntCommandName>RemoveAllShipUnits</IntCommandName>
</IntCommand>
```

To remove only non-permanent ship units (where IsPermanent = 'N'):

```
<IntCommand>
   <IntCommandName>RemoveNonPermanentShipUnits</IntCommandName>
</IntCommand>
```

To remove orphaned ship units, use the command below. This specifies that the ship units that have been removed from the shipment via the DR transaction code should be deleted if no other shipments refer to them. Without this command, those ship units are left in the system and can later be added to other shipments.

```
<IntCommand>
   <IntCommandName>DeleteOrphanedShipUnits</IntCommandName>
</IntCommand>
```

Alternative Interfaces For Updating Ship Units

For alternatives to using this interface to update ship unit information, see SShipUnit and TransOrder.

Tips For Shipments As Work or SAWs

Element	Description

Element	Description
ShipmentHeader2	In Shipment/ShipmentHeader2, the most important element is ShipmentAsWork and it should almost always be set to "Y". The exception is when: 1) The shipment is new AND 2) There is at least one release associated with the shipment OR the shipment.ShipmentHeader2. AutoGenerateRelease = "Y" To avoid confusion, set the Perspective element to "B." If Perspective is not specified, it will default to "B". All other elements in this branch are completely optional from a schema and business perspective.
ShipmentHeader, use correct rate	In the Shipment/ShipmentHeader, if you can get them, it is good to provide the RateOfferingGID and the RateRecordGID. This helps Oracle Transportation Management use the rate for the service provider that actually took the load. If you can't get information for these elements, the ServiceProviderGID would be the next best thing to use.
sPlannedTimeFixed	If you want to send in old shipments (past dates) and want Oracle Transportation Management to rate with correct rates (pertaining to correct effective/expiration dates), then you may want to use the Shipment/ShipmentStop/ArrivalTime/EventTime/IsPlannedTimeFixed flag set to "Y". You only need to insert the ArrivalTime element at the first stop (only) and that the date here should be the same as the StartDate of the shipment.
SEquipment	Oracle Transportation Management requires at least one SEquipment object. When you insert a new SAW, Oracle Transportation Management creates a default SEquipment if you do not provide one. If there are several ship units, the same (created equipment) is specified for each ship unit. The only thing really required in SEquipment is the SEquipmentGID. To avoid problems later, include the SEquipment element and set the SEquipmentGID to the same value as the ShipmentGID. This makes it easier to identify and manage the SEquipment, if there is ever a need in the future to specify multi-equipment, or update the shipment with additional ship unit information. **Note:** When you update a SAW with a new ship unit, you must include the SEquipmentGID. Oracle Transportation Management cannot create one for you.
TransOrder	The Shipment/TransOrder element is outbound only so you cannot include it. Any TransOrderHeader info should be specified in the Release/TransOrderHeader element.

Element	Description
Locations	When specifying source and destination locations in the Shipment/Release/ShipFromLocationRef and Shipment/Release/ShipToLocationRef elements, refer to locations already defined in Oracle Transportation Management instead of defining new ones (using Shipment/Release/ShipFromLocationRef/LocationRef/Location). This saves you the effort of providing all the Location elements. To refer to existing locations, use the Shipment/Release/ShipFromLocationRef/LocationRef/LocationGID element.
ShipUnits	Set ShipmentHeader2/AutoGenerateRelease to "Y" to avoid having to populate both Shipment/Release/ShipUnit and Shipment/ShipUnit. **Note:** Oracle Transportation Management generates an error if AutoGenerateRelease is set to "Y" and you still include the Shipment/Release element.

Order-Centric Modifications

Most modifications via this interface are based around shipment ship units (SShipUnit). In these cases, all weight, volume, quantities, and rating are based on shipment ship units. If you would like to modify shipments based on order information, you can do so by using the following sub-elements in the ShipmentHeader element:

- ShipmentModViaOrderLine
- ShipmentModViaOrderSU

When these two elements are used in the ShipmentHeader, the following logic will be used instead of the standard Shipment Interface logic:

1. Oracle Transportation Management will only interact with the order line level or order ship unit information instead of the shipment ship unit level information.
2. The logic addressing shipment modifications will change the number of order ship units involved and allocate the delta in the ship unit counts across multiple ship units.
3. The modified order ship unit count will be properly propagated and the related business objects (shipments and order movements) will be updated across legs.
4. The modified gross weight and volume will be updated per ship unit. This would then be reflected in the shipment total gross weight and volume, which impacts the shipment cost. This should only be applied when the AffectsCurrentLegOnly element is set to 'N'.

Both of these elements will only be included once on the shipment. There is no need to repeat this data for both the pickup stop and the delivery stop. Since the Shipment Interface is defined the same on the inbound and the outbound, you can only specify one way for the modification to happen, either at the order line level or the ship unit level.

ShipmentModViaOrderLine

The ShipmentModViaOrderLine element will contain all of the counts, weights, and volumes for that order (order release, order release line, or order base) that is being shipped on this shipment across all the shipment ship units.

ShipmentModViaOrderSU

The ShipmentModViaOrderSU element, Oracle Transportation Management will loop through all the ship units that are on the shipment that have the same order ship unit GID (ob_ship_unit_gid or or_ship_unit_gid).

The inbound XML will accept this data into Oracle Transportation Management when you are doing a modify transaction. When the integration brings in this modification it will call business logic that will apply allocation rules and perform the appropriate updates.

Data Requirements

To send shipments and perform planning actions, you must make decisions about the way you want Oracle Transportation Management to perform certain actions.

Sending Shipments (Shipment as Work)

Send shipments that do not have orders associated with them to Oracle Transportation Management for processing using the Shipment interface. This type of shipment is known as a shipment as work or manual shipment; it can include order level information, but not necessarily. These shipments are not bundled, re-consolidated, or re-sequenced.

A shipment as work must have at least one pickup and one delivery location. A shipment as work is not associated with an itinerary.

> **Note:** To indicate that the shipment you are sending to Oracle Transportation Management is a shipment as work, enter Y in the ShipmentAsWork element.

When a Shipment as Work is received, Oracle Transportation Management can be set to automatically perform certain actions defined in public workflow agents in the Agent Manager.

To ensure best possible performance, you should let Oracle Transportation Management process your actual shipments in parallel. To do this, either send only one actual shipment per transmission or follow these steps:

- In the TransmissionHeader, set IsProcessInSequence to N.
- Send all the actual shipments in one Transmission.

See the Shipment Manager help for a detailed description of manual shipments.

If you insert a new shipment and omit the end_date, Oracle Transportation Management sets the end_date to the same date as the start_date.

AllocationBase Interface

The AllocationBase outbound interface sends order release allocation cost information to an external system. Allocation is a method for dividing the cost of a shipment among its order releases based on the line items and ship units on the shipment. An allocation transmission consists of an OrderRelease GID and cost information. You can send allocation information about planned and actual shipments.

You can create an agent that send the AllocationBase interface using the agent action Send Allocation Interface. You can also send this interface from the Invoice Manager, Shipment Manager, and the Order Release Manager.

You must setup allocation to follow the rules for allocating shipments used at your organization.

Process Allocations

Instruct Oracle Transportation Management to send an allocation transmission using the Process Manager:

- Issue Allocation
- Bulk Allocation

Accrual Interface

The Accrual interface is used to transmit the allocated freight cost accrued or paid against orders. This can be used to communicate changes or differences in an order's allocated freight cost to other external systems. These differences, for example, could be used to establish liabilities in an accounting or general ledger system.

When the ALLOCATION GENERATES ACCRUALS parameter is set to TRUE and the shipment status is ACCRUAL_ALLOWED, the allocation logic will generate an accrual record. The accrual record contains the difference between the current allocated freight cost and the previously transmitted freight cost. The delta between the two is used because a single order may be on multiple shipments, which are approved for payment in different time periods. These accrual records are sent as part of this interface.

The Accrual interface is supported on the outbound only. It can also be sent via the user interface in the Process Manager, Send Integration page.

ExchangeRate Interface

You can send exchange rates from and to Oracle Transportation Management through this interface. You might want to do this if the default IMF feed does not work for your requirements.

Insert New Exchange Rate into Oracle Transportation Management

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

5. Set the TransactionCode to I.
6. See the Exchange Rate Manager online help for a description of the fields.
7. See the ExchangeRate Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your exchange rate definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Modify Exchange Rate in Oracle Transportation Management

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to U.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the Exchange Rate Manager online help for a description of the fields.
4. See the ExchangeRate Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your modified exchange rate definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Delete Exchange Rate in Oracle Transportation Management

Required Data

You must know the ExchangeRateGID of the exchange rate definition you want to delete.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D.
2. Enter the ExchangeRateGID.

Transmission Results

Oracle Transportation Management deletes your exchange rate definition in the database.

Error Messages

If you get a TransmissionReport that reads "CAUGHT THE FOLLOWING EXCEPTION WHILE PROCESSING TRANSACTION:", make sure you are not trying to delete an exchange rate definition that is used elsewhere in Oracle Transportation Management.

If you receive a TransmissionReport, check for integration messages.

Send Exchange Rate from Oracle Transportation Management

Setup

You must have created the External System you want to send to.

How To Send the Transmission?

- In the Exchange Rate Manager.
- In the Process Manager, Send Integration Page.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified for the exchange rate definition.

Transmission Results

Error Messages

Invoice Interface

Service providers and other third parties send invoices for payments on shipments to Oracle Transportation Management through the Invoice interface. The invoice transmission contains details about the payment and the specific transportation activity for which payment is owed.

Invoices can be automatically matched to shipments based on the Service Provider ID and Shipment Reference Number fields. If more than one shipment is found for an invoice, the invoice must be reviewed manually and assigned to a shipment, it must be rejected. After an invoice is approved, a voucher gets created. A voucher represents what the planner agrees to pay for the shipment.

Use the Financials managers to create and modify invoices and customer bills.

In some cases, you may need to send the Invoice interface outbound. This is true when sending a bill to yourself for internal invoice or billing purposes. When the Invoice interface is used outbound, then it will include all shipment details, as well as any associated order information.

Consolidated Invoices

1. You must send each invoice, parent and child, as a separate transaction.
2. Parent invoices must enter Oracle Transportation Management before any child invoices.
3. Child invoices may be sent inbound referencing a parent in one of two ways:
   - Populate the invoice number on the child to be that of the parent and integration will lookup the parent id based on the invoice number.
   - Populate the parent invoice ID on the child.

ItemMaster Interface

The ItemMaster interface provides a way to transmit item information to Oracle Transportation Management. Items represent the freight being shipped. The ItemMaster transmission includes packaging elements, elements that describe the item, and any NMFC, STCC, SITC, or HTS codes that apply. In addition, a general ledger GID or accessorial charges can be included.

> **Note:** Item information must be in Oracle Transportation Management before it can accept TransOrders that reference the item.

Itinerary Interface

The Itinerary interface is part of the GLogXMLElement element. It is used to define the path between two locations and specifies the constraints for building the shipment. This element is supported on the inbound only.

For details on the elements, both required and optional, as well as their format and descriptions, please view the XML Element Dictionary.

Job Interface

Use the Job Interface to send data that addresses how logistics service providers and freight forwarders manage the services they provide within Oracle Transportation Management. A job offers a workspace that brings together the objects and activities required of them, including:

- The ability to group all objects related to a job and perform existing functions/actions against those objects, including buy shipments, sell shipments, non-freight related charges, and customer bills.
- The ability to manage jobs from various perspectives depending on responsibility. For example, export, import, both, or consolidations.
- The ability to manage settlement functions at the job level, including profitability, expenses, revenues, and billing.

Note: The interface is supported on the outbound only.

The primary business objects in the Job Interface are:

1. Order releases
   - For each order release related to the job, JOB_ORDER_RELEASE_JOIN will be added to the XML. Although orders are not required to create a job, at least one order should be related to the job to send out the interface.
2. Buy side costs
   - The Buy Side Costs wrapper element contains two sub elements, Buy Shipments and Buy Allocation.
   - Buy shipments: Select all related orders. For each order, select all related buy shipments where the shipment job GID equals null or it equals the current job GID. There may be zero or more buy shipments.
   - Buy allocation: Each allocation will be selected for each order, where the allocation Shipment Job GID is equal to the current job or it is equal to null. There may be zero or more allocations.
3. Sell side costs
   - The sell side costs works exactly the same as the buy side, except the selection criteria is based on sell side perspective.
4. Bills
   - Each customer bill related to the job will be included in the XML. Bills can be found in the JOB_BILL table. Zero or more bills are required.
   - Because this interface can potentially be large, redundant data has been reduced across multiple data elements included in the interface. This includes:
   - The ability to only include the Order Release GID in the shipment, allocation, and bill elements.
   - The ability to only include the Shipment GID in the allocation and bill elements.

Location Interface

The Location element specifies a location.

Insert New Location into Oracle Transportation Management

Required Data

If your location is associated with a corporation, you need to have that corporation defined.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

A location may be transmitted to Oracle Transportation Management as a stand-alone transmission, or embedded in other transmissions. Sending a location embedded in another transmission can save time.

What Data Goes into the Transmission?

1. Set the TransactionCode to I or II.
2. See the Location Manager online help for a description of the fields.
3. See the Location diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your location definition in the database.

If you send a location to Oracle Transportation Management with a new child domain, Oracle Transportation Management adds the child domain to your database automatically.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Modify Location in Oracle Transportation Management

Required Data

Setup

You control validation of incoming location transmissions with the glog.integration.validation.locationinterface property.

What Data Goes into the Transmission?

1. Set the TransactionCode to U.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the Location Manager online help for a description of the fields.
4. See the Location diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your modified location definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Delete Location in Oracle Transportation Management

Required Data

You must know the LocationGID of the Location you want to delete.

Setup

You control validation of incoming location transmissions with the glog.integration.validation.locationinterface property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D.
2. Enter the LocationGID.

Transmission Results

Oracle Transportation Management deletes your location in the database.

Error Messages

If you get a TransmissionReport that reads "CAUGHT THE FOLLOWING EXCEPTION WHILE PROCESSING TRANSACTION:", make sure you are not trying to delete a location that is used elsewhere in Oracle Transportation Management.

If you receive a TransmissionReport, check for integration messages.

Send Location from Oracle Transportation Management

Required Data

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

- In the Location Manager.
- In the Process Manager, Send Integration Page.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified for the location.

Transmission Results

Error Messages

Mileage Interface

Use the Mileage interface to transmit distance information to Oracle Transportation Management. The mileage interface identifies the distance between the points in a lane. Lanes may contain specific or general geographic information, such as:

- Zip code to zip code
- City to city
- State to state
- Zip code to city
- City to state
- Address to address
- City to address
- State to address

As a result, the mileage data may enter Oracle Transportation Management with different level of precision with regard to specific locations.

Mileage information may be changed whenever necessary, using the Mileage interface, or by using Power Data. The mileage interface may be used to create new lane information, or update existing information or when planning an order.

Release Interface

The Release interface transmits order release information to and from Oracle Transportation Management. An order release represents the shippable portion of a transportation order. After you send a release, Oracle Transportation Management automatically creates an order base for it. A release from a Transportation Order corresponds to the particular TransOrder Lines, which share common transportation requirements.

TransOrders represent the demand for transportation services, and consist of header information and either line items or ship units, but not both.

Note: The TransOrder element is only supported on the outbound Release interface.

An order release contains the following information:

- Order release ID that is automatically generated by Oracle Transportation Management.
- Order release name and type.
- Order base ID that references the base order from which the order release was created.
- Source and destination locations.
- Early/late pick up dates.
- Assigned or fixed itinerary.
- Current status.
- Package or non-package data attributes.

Release Method

Populate the ReleaseMethodGid to tell Oracle Transportation Management how to release ship units. The ReleaseMethodGid (also called Order Configuration) tells Oracle Transportation Management how to build ship units from order lines, or how to calculate ship unit information if ship units are entered. If ReleaseHeader/ReleaseMethodGid is not populated, the default can be property controlled. See the Order Management Guide for more information.

Business Number Generator (BNG)

You can send a transportation order to Oracle Transportation Management, without the entering values in the TransOrder GID, Ship Unit ID, or Order Release ID elements in the XML Transmission. Oracle Transportation Management generates values for these fields based on the default business number rule in place when the order comes into the system. You can set up the BNG to create numbers that fit your needs.

ReleaseInstruction Interface

The ReleaseInstruction interface allows you to select specific line items and ship units, and release specific quantities of them for a particular order base.

Included in this interface are detailed instructions for creating the order release.

RemoteQuery Interface

Use remote queries to request certain kinds of information from Oracle Transportation Management.

> **Note:** When you create a RemoteQuery, enter QUERY in the TransmissionType element.
> `<TransmissionType>QUERY</TransmissionType>`

ShipmentQuery

Use the ShipmentQuery element to request information about a shipment. Send Oracle Transportation Management the shipmentID, and receive a shipment transmission that includes all the details associated with the shipment.

RIQQuery Interface

Use the RIQQuery element to request rate information for a shipment. Ask for rates based on service provider, transportation mode, quantity, locations, and arrival or departure dates. For example, you might request a list of carriers, show best rates, or fastest routes.

For air rates, Oracle Transportation Management looks for known shippers by default and therefore returns cargo as well as passenger flights.

If the AvailableBy element is not passed, the current date is used.

> **Note:** For the remote query to work properly, rates, locations, itineraries, and all other shipment-related information must be fully loaded in Oracle Transportation Management.

> **Note:** You must create an external system and enter a correct IP address in order to pass authentication.

Transmission ReportQuery

If you send a transmission to Oracle Transportation Management but do not get a TransmissionReport back, use the TransmissionReportQuery element to request the missing TransmissionReport.

> **Note:** This only makes sense if glog.integration.TransmissionReport has been set to yes.

RemoteQueryReply Interface

The RemoteQueryReply contains Oracle Transportation Management's response to your RemoteQuery. You use the RemoteQuery interface to send remote queries to Oracle Transportation Management.

Oracle Transportation Management automatically sends its RemoteQueryReply to the IP:port number of the originating RemoteQuery transmission.

If you are trying to send more than one RIQ into Oracle Transportation Management using the RemoteQuery interface, then you need to set the glog.integration.RemoteQuery.WrapReplyInTransmission property to 1. Since the RemoteQueryReply interface only sends one reply back, it will drop the additional RIQs. With this property set to 1, it will wrap the reply in the XML transmission which will send all the RIQs back in the reply.

ServiceTime Interface

Use the ServiceTime interface to transmit the time it takes to go from one point of an XLane to the other. You can use this information for shipment planning.

ServiceTime transmissions can be sent as part of a Mileage interface transmission. Service time may also be transmitted to Oracle Transportation Management as a GLogXMLElement. When identified as part of a Mileage element, Oracle Transportation Management ignores the TransactionCode and XLaneRef elements.

ShipmentStatus Interface (INE)

Service providers and other third parties transmit shipment event information to Oracle Transportation Management with the ShipmentStatus interface. Shipment status information describes activity on shipments and shipment groups. For example, you can use shipment statuses to determine whether a shipment is running on time or whether it is late. In addition, you can send vessel and flight information, and equipment service provider and order information. Based on shipment status information it receives, Oracle Transportation Management can re-plan a shipment.

You can also send received ship unit quantities with this interface using the SStatusShipUnit element.

Insert New Shipment Status into Oracle Transportation Management

Required Data

Before you can send ShipmentStatus transmissions to Oracle Transportation Management, you must set up the following:

- User accounts for service providers in Oracle Transportation Management
- Shipment Status Codes
- Shipment Status Reason Codes
- Shipment Event Groups
- Shipment Reason Groups

- Corporations for service providers

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

What Data Goes into the Transmission?

1. To ensure that Oracle Transportation Management processes multiple ShipmentStatus transactions in the order you intend, set IsProcessInSequence to Y in the TransmissionHeader.
2. Identify which object (shipment, shipment group etc) the status applies to. Set StatusLevel, ShipmentStatusType, ServiceProviderAlias, and ShipmentRefnum or IntSavedQuery.
3. ShipmentStatusType must be set to one of Shipment or ShipmentGroup. Note that it is case sensitive. The integration logic assumes that it is a ShipmentGroup if the value is not matched to Shipment.
4. Optionally, identify which equipment the shipment status refers to with SStatusSEquipment element.
5. Include the time when the event occurred. To be sure that Oracle Transportation Management can interpret the time correctly, include the TimeZoneGID element. Alternatives to doing this is:
   - If you cannot include the TimeZoneGID, Oracle Transportation Management can set the time zone to the time zone of the Location where the event occurred.
   - If you cannot do this either, set the TimeZoneGID to Local. In this case, Oracle Transportation Management saves and displays the event date as entered, ignoring user preferences.
6. Enter your status information. In some cases, shipments can only have events added to them if they are of a certain status.
7. Identify at what SSStop (number or location name) the event (shipment status) occurred. The LocationID = Location Reference Number and the LocationRefnumQualifierGID = Location Reference Qualifier in the Location Manager.
8. If you have a Shipment Agent Type with the Recalc Estimated Stop Times action, then you must include a RATE_GEO element for Oracle Transportation Management to be able to recalculate your estimated stop times and/or re-drive your shipment. If you omit the RATE_GEO element, Oracle Transportation Management only resets the stop times you provide.
9. See the ShipmentStatus Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your information in the database.

Agents might have been set up to act upon receiving certain shipment statuses.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Send Shipment Status from Oracle Transportation Management

You can forward a received ShipmentStatus transmission to an external system with an agent. See the agent action called SEND SHIPMENT STATUS XML.

Match Events to an Object

You can use one of these methods:

- If you need the object (shipment or shipment group) to match many reference numbers, use IntSavedQuery.
- If you need the object to only match one out of a set of reference numbers, use ShipmentRefnums, or ShipmentGroupRefnums.

IntSavedQuery

If you specify the IntSavedQuery element, only that query is applied. You can define a query to search for shipments or shipment groups that shipment status applies to. To do this, set:

- IntSavedQueryGID to specify which query you want to use. If the query you specify here does not return any results, Oracle Transportation Management generates an error message. No other queries are applied. You must have created this query in Power Data beforehand.
- IntSavedQueryArg to arguments that can be referred to in the queries. For example, BM=YELLOW-0000007. If you omit this element, your IntSavedQueryGID must point to a query that uses XPath instead.
- IsMultiMatch to N to forbid multiple records to be returned from the query. If your query happens to return multiple records, Oracle Transportation Management generates an error message.

Refnums

If you omit the IntSavedQuery element, Oracle Transportation Management tries to match your shipment status with:

- The ShipmentRefnum elements to the shipment_refnum table in the database.
- The SSEquipment/ EquipmentIdentificationNum element to the S_Equipment.Equipment_Number field in the database
- The standard integration saved query INT_SHIPMENT_STATUS_GID_1
- The standard integration saved query INT_SHIPMENT_STATUS_GID_2

You can optionally enforce a rule that a give shipment may have only one shipment reference number with a given qualifier. The update_flag column in the shipment_refnum_qual table indicates if the rule is in effect or not. The valid values for the update_flag are:

- UPDATE_OK: Only one value is allowed for a give qualifier, the value of which can be modified.
- UPDATE_NOT_OK: Only one value is allowed for a give qualifier, the value of which cannot be modified.
- MANY: a given shipment can have multiple values for the same qualifier.

Match Events to a Shipment Stop

For Oracle Transportation Management to match an event to a stop on a shipment, you must include the SSStopSequenceNum element.

Another way of matching event to shipment stop is to include the LocationID where the event occurred and the LocationRefnumQualifierGID in SSStop/SSLocation. This only works if you have enabled this feature in your glog.properties file. As long as Oracle Transportation Management can match your LocationID to a stop number, your shipment status saves as if you had supplied a stop number.

Note: If Oracle Transportation Management cannot match the event to a location, Oracle Transportation Management still saves the information but not for a specific stop and only as informational. You can also have Oracle Transportation Management send you a TransmissionReport if the LocationID is missing altogether (controlled by glog.properties). Oracle Transportation Management also set the time zone for the event to local.

Correspondingly, if the event is not related to a shipment stop to begin with, Oracle Transportation Management saves the event as informational with a local time zone.

A single stop related shipment event can be applied to multiple shipments, regardless of whether their stop numbers or location IDs are the same. This will allow for situations where you want to apply a single shipment stop event to stop 2, but stop 2 of shipment 1 and stop 1 of shipment 2 are both Philadelphia. Stop related events are applied to all the shipments specified in the ShipmentStatus interface. To work successfully, the ShipmentStatus XML must include an IntSavedQuery element that will return two shipments. Logically, this is similar to having specified the ShipmentStatus message multiple times in the Transmission XML.

ShipStop Interface

The ShipStop interface is used to modify stop related information for a shipment.

It is an easier alternative to updating shipment stop information than doing it directly through the Shipment interface.

There are four main elements to the ShipStop interface:

- SendReason: Used to indicate the reason the notification/data/information is being sent to the external system.
- IntSavedQuery: See online help for details.
- ShipmentGID: It is a shipment global identifier. It uniquely identifies a shipment within the Oracle Transportation Management.
- ShipmentStop: They are the pickup and delivery points for a shipment. Ship units and lines represent the freight and packaging carried on a shipment from one stop to another. Sub-elements of the ShipmentStop element mirror the fields found in the Shipment Stops Manager.

TenderOffer Interface

Tender a planned shipment to a service provider.

The service provider's reply to the TenderOffer is sent through the TenderResponse interface.

Send New TenderOffer from Oracle Transportation Management

Required Data

- Define an external system where Oracle Transportation Management should send the TenderOffer transmissions.
- Associate the contact of your service provider with the external system.
- Choose HTTPPOST or EDI as the main communication method for the contact of your service provider.
- Choose HTTPPOST or EDI as the main communication method for the logistics contact on the shipment. Choose the same communication method as for the contact of your service provider.

- Set up shipment planning and processing.

Setup

Set parameters in Power Data that determine how many tender offers are automatically sent for single shipment and how much time you want to allow before a tender times out.

How To Send the Transmission?

You can either use the Tender action in the Shipment Manager, or create an agent.

What Data Goes into the Transmission?

Transmission Results

When you tender a shipment, Oracle Transportation Management performs the following actions:

- Sends notice of the tender offer to the service provider.
- Starts a timer that defines how long the service provider has to respond to the tender before it is withdrawn.
- Updates the shipment status to SECURE RESOURCES_TENDERED.

If the service provider declines the tender offer, Oracle Transportation Management automatically re-tenders the shipment and re-initiates the tender process by sending a tender transmission to a new service provider. If no other service provider is available, the shipment status is updated to SECURE RESOURCES_NO_RESOURCES.

If the tender offer times out, Oracle Transportation Management notifies the service provider of the tender withdrawal, and re-tenders the shipment to another service provider. A tender offer times out when the amount of time established for a service provider to respond to a tender expires.

Error Messages

Cancel TenderOffer

Required Data

You must have sent a new TenderOffer.

Setup

How To Send the Transmission?

Shipment Manager, Withdraw Tender action

What Data Goes into the Transmission?

Oracle Transportation Management sets the TransactionCode to D.

Transmission Results

Error Messages

Batch Process Tender Offers

Oracle Transportation Management can use a batch process to tender multiple shipments.

TenderResponse Interface

Responds to a TenderOffer.

Oracle Transportation Management sends out a TenderOffer asking if a particular service provider wants to carry a shipment. Service providers send back a TenderResponse stating whether they agree to carry the shipment.

Insert New TenderResponse into Oracle Transportation Management

Required Data

- You need to know the ITransactionNo from the TenderOffer you are accepting or declining.
- See the online help topic: Configuring Online Booking and Tendering.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Enter the ITransactionNo from the TenderOffer.
2. Set the ActionCode to A (Accept) or D (Decline).
3. See the online help for a description of the fields.
4. See the TenderResponse Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

When you accept a tender, Oracle Transportation Management performs the following actions:

5. Updates the shipment status to SECURE RESOURCES_ACCEPTED. For a pickup notification, Oracle Transportation Management sends pickup information to the service provider including where and when the pickup must occur.
6. Cancels the tender time-out alarm. The tender time-out alarm defines the amount of time in which a service provider can respond to a tender.
7. You can have Oracle Transportation Management send notification to involved parties when a tender is accepted.

If the tender is declined, Oracle Transportation Management performs the following actions:

8. Updates the shipment status to SECURE RESOURCES_DECLINED.
9. Creates a Response to Tender Log entry for the shipment.
10. Identifies the Declined Route so that the same service provider is not offered the shipment a second time.
11. Accesses re-tender rules. The re-tender log contains user-defined rules for how alternate service providers are evaluated and how many times a shipment gets tendered automatically by Oracle Transportation Management.

Error Messages

If you receive a TransmissionReport, check for integration messages.

TransmissionReport Interface

Lists why the transmission, sent to Oracle Transportation Management, was not processed successfully.

Send Transmission Report from Oracle Transportation Management

Required Data

An external system must have sent a transmission to Oracle Transportation Management.

That transmission must contain a Transmission/TransmissionHeader/AckSpec with a return URL or email address.

Setup

glog.integration.TransmissionReport controls what kind of errors trigger Oracle Transportation Management to send a TransmissionReport. You can override these settings with the AckSpec element of your transmission.

You can speed up the generation of TransmissionReports by omitting the TransmissionSummary element with the glog.integration.transmissionreport.includesummarydetail property.

How To Send the TransmissionReport?

Oracle Transportation Management sends the TransmissionReport after the transmission has been processed.

Oracle Transportation Management sends the TransmissionReport via either email or HTTPPOST depending on what the AckSpec element in the TransmissionHeader of the original transmission specifies.

What Data Goes into the TransmissionReport?

The TransmissionReport consists of a transmission number identifying the transmission, a sender number used for acknowledgement, as well as integration log messages and summary information. The TransmissionReport details the errors that must be corrected before the data can be re-transmitted. For example, if the transmission uses a location that is not already in the database, the transmission report would include a foreign key error.

Transmission Results

Error Messages

If you are not getting a TransmissionReport, check the following properties:

```
glog.integration.transmissionReport.readResponse
glog.integration.TransmissionReport
```

TransOrder Interface (INO)

Create, modify, or delete order information through the TransOrder interface.

Insert New Order and Release Order Line

This procedure shows you how to:

- insert a new order
- release all or part of an order line
- build shipments from the order release

Required Data

To send an order to Oracle Transportation Management, certain information related to the order must already exist in Oracle Transportation Management. For example, you must have a valid itinerary, rate, locations, and so on.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to I. A transaction code of UI or IU works too.
2. If you do not want to enter values for the TransOrderGID, TransOrderLineGID, ShipUnitGID, or OrderReleaseGID elements, you can have Oracle Transportation Management automatically generate GIDs. Automatic generation of GIDs only works for a transaction code of I.

   Note: If a transaction code of IU is used, then a TransOrderGID must be provided.

3. Populate the ProcessingCodeGID to tell Oracle Transportation Management how to plan the shipments from the order release.
4. Populate the TransOrderLineDetail element, including the PackagedItemCount, WeightVolume/Weight, and WeightVolume/Volume under TransOrderLineDetail/TransOrderLine/ItemQuantity/ to specify the order lines. You can set all but one of them to 0, if your setup uses the same kind of quantity to release.
5. Set TransOrderLineDetail/TransOrderLine/ItemQuantity/IsShippable = N.
6. Populate the amount to release in the TransOrderHeader/ReleaseInstruction/QuantityToRelease element.

   If you omit the ReleaseInstruction, set TransOrderLineDetail/TransOrderLine/ItemQuantity/IsShippable = Y to have Oracle Transportation Management create an order release for all of your order lines.
7. If your order base is coded in a format other than GLogXML you need to transform your TransOrder transmission into the GLogXML schema, you can use Oracle Transportation Management's transform feature to do this.
8. If you want to change the level of validation for this transmission, you can include a processing instruction to set the desired level. By default, TransOrder validation is turned on.
9. See the online help for a description of the fields.
10. See the TransOrder Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

11. Oracle Transportation Management receives your transmission and starts to process it internally.

12. Oracle Transportation Management starts the public Order Base - Insert agent.

    If the current date is outside the effective date/expiration date window of your TransOrder, the agent cannot create order releases. You must release the TransOrder via the process manager. There you can release all orders which have release instructions, but whose release has not been processed.

    If you use the UI or IU transaction codes and the record exists already, Oracle Transportation Management starts the public Order Base - Modify agent instead.
13. Oracle Transportation Management raises events that in turn can trigger Notifications to be sent.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Insert New TransOrder and Release ShipUnit

Auto-releasing ship units is consistent with order base lines. This procedure shows you how to:

- insert a new order
- release all or some ship units on the order base
- build shipments from the order release

Required Data

To send an order to Oracle Transportation Management, certain information related to the order must already exist in Oracle Transportation Management. For example, you must have a valid itinerary, rate, locations, and so on.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to I. A transaction code of UI or IU works too.
2. If you do not want to enter values for the TransOrderGID, ShipUnitGID, or OrderReleaseGID elements, you can have Oracle Transportation Management automatically generate GIDs. Automatic generation of GIDs only works for a transaction code of I.

   Note: If a transaction code of IU is used, then a TransOrderGID must be provided.
3. Populate the ProcessingCodeGID to tell Oracle Transportation Management how to plan the shipments from the order release.
4. Populate the TransOrder/ShipUnitDetail element.
5. To be able to track your ship unites as they propagate through Oracle Transportation Management as order release ship units and shipment ship units, you might want to include a unique ID in the ShipUnitDetail/ShipUnit/ShipUnitContent/ItemQuantity/ItemTag1 element. Also, there is a TransOrderShipUnitGID element in Release/ShipUnit that can help you track ship units.
6. Set ShipUnitDetail/ShipUnit/IsShippable to Y to have Oracle Transportation Management create an order release for all your order base ship units.

7. If you omit the IsShippable element or set it to N, you need to populate the amount to release in the TransOrderHeader/ReleaseInstruction/QuantityToRelease element. With this option, you can specify the number of ship units to be released in the ReleaseInstruction/ShipUnitReleaseCount element.
8. You can override all dates and locations from the ShipUnitDetail with other settings in the ReleaseInstruction.
9. If you want to change the level of validation for this transmission, you can include a processing instruction to set the desired level. By default, TransOrder validation is turned on.
10. See the Order Base Manager online help for a description of the fields.
11. See the TransOrder Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

1. Oracle Transportation Management receives your transmission and starts to process it internally.
2. Oracle Transportation Management starts the public Order Base - Insert agent.
3. It finds the unprocessed release instructions with a release date <= the current date.
4. If the current date is outside the effective date/expiration date window of your TransOrder, the agent cannot create order releases. You must release the TransOrder via the process manager. There you can release all orders which have release instructions, but whose release has not been processed.
5. If you use the UI or IU transaction codes and the record exists already, Oracle Transportation Management starts the public Order Base - Modify agent instead.
6. Oracle Transportation Management raises events that in turn can trigger notifications to be sent.

Modify Order Base With Lines

In this scenario, you can just update the information in an order base or you can update and release the full amount specified for the TransOrderLine.

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation.orderinterface property.

What Data Goes into the Transmission?

1. Set the TransactionCode to U.

   A transaction code of UI or IU works too.
2. If your update is to delete only a couple of fields in the OrderBase, use the Value to Null Field symbol.
3. If you want Oracle Transportation Management to release all your order lines, set IsShippable = Y.

   With IsShippable=Y, you should omit the ReleaseInstruction element, Oracle Transportation Management releases the full weight, volume, or count (depending on glog.properties). If you set IsShippable=Y and include a ReleaseInstruction, Oracle Transportation Management releases your order twice. One full order release based on the parameter in glog.properties and another order release based on the ReleaseInstruction element.

4. If you want to change the level of validation for this transmission, you can include a processing instruction to set the desired level. By default, TransOrder validation is turned on.
5. See the Order Base Manager online help for a description of the fields.
6. See the TransOrder Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.
7. To update date fields with NULL values, submit a value of '~' in the date element(s) of the inbound TransOrder XML.

Transmission Results

1. Oracle Transportation Management receives your transmission and starts to process it internally.
2. Oracle Transportation Management starts the public Order Base - Modify agent.

   If you use the UI or IU transaction codes and the record does not exist already, Oracle Transportation Management starts the public Order Base - Insert agent instead.
3. Oracle Transportation Management raises events that in turn can trigger Notifications to be sent.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Modify ShipUnits

There are three ways to update the ship unit information (quantities, weights, volumes, etc) on a shipment via integration:

- Use ActualShipment. This interface provides complete control of all the fields in the Shipment.
- Use SShipUnit.
- Send another TransOrder with the IsUpdateShipmentOnly element.

The TransOrder interface together with the IsUpdateShipmentOnly element supports uploading a slightly modified TransOrder and has it update only the Shipment/SShipUnit. IsUpdateShipmentOnly indicates that the TransOrder should update the shipment only, and not the order base information.

> **Note:** To update date fields with NULL values, submit a value of '~' in the date element(s) of the inbound TransOrder XML.

Using the IsUpdateShipmentOnly element can help you reduce the need to implement a separate SShipUnit or ActualShipment interface. The use of this flag with the TransOrder interface is restricted as follows:

- The original order base should have been created using the ShipUnitDetail (not the TransOrderLineDetail).
- The information you can update is restricted to the SShipUnit element. The TransOrderHeader is ignored, and none of the other Shipment related information is updated.
- The specific S_Ship_Unit(s) to be modified are identified by using the ShipUnitGID in the new TransOrder and searching for the related Release/ShipUnit (via the OB_SHIP_UNIT_GID on SHIP_UNIT table) and then the Shipment.ShipUnit(s) (via the SHIP_UNIT_GID field in the S_SHIP_UNIT table). The search requires those reference pointers to exist.

Delete Orders

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation.orderinterface property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D.
2. If you do not know the GID of the record you want to delete, you can use integration saved queries instead.
3. If you want to change the level of validation for this transmission, you can include a processing instruction to set the desired level. By default, TransOrder validation is turned on.
4. See the Order Base Manager for a description of the fields.
5. See the TransOrder Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

1. Oracle Transportation Management receives your transmission and starts to process it internally.
2. Depending on what kind of record you are deleting an agent might start. For example, if you are deleting an Order Base, the public Order Base - Delete agent starts.
3. Oracle Transportation Management raises events that in turn can trigger notifications to be sent.

Error Messages

You cannot delete an order that is assigned to a shipment after a service provider accepts a tender on the shipment. If you try to do this, Oracle Transportation Management might send a TransmissionReport stating the problem.

If you receive a TransmissionReport, check for integration messages.

Bulk Plan Orders

Required Data

You must create a saved query that points out the order releases you want to include.

Setup

You control validation of incoming transmissions with the glog.integration.validation.orderinterface property.

What Data Goes into the Transmission?

If you can keep all your TransOrders within one transmission follow these steps:

1. Set IsProcessInSequence=N.

   This ensures maximum performance because Oracle Transportation Management can process TransOrders belonging to different order bases in parallel.
2. Include all TransOrders that should be bulk planned.
3. Create an order release for every TransOrder either with IsShippable=Y and omit the ReleaseInstruction, or with IsShippable=N and include a ReleaseInstruction.
4. Include a Topic element as the last element in the transmission to start the bulk planning. Set TopicArgName to 'savedQuery' and TopicArgValue to a Query_Name. The saved query must point out the Order Releases you want to include.
5. In the GLogXMLElement holding the Topic element, include a ProcessInfo element with WhenToProcess=END_OF_TRANSMISSION.

   This tells Oracle Transportation Management to wait to start the bulk planning until the end of the transmission.

   Note: Oracle Transportation Management plans all order releases that match the saved query, not just the ones within the transmission.

   Note: The Topic element must be the last element in the transmission. If it is not, Oracle Transportation Management will plan incorrectly.

If you cannot keep all your TransOrders within one Transmission follow these guidelines:

1. For every transmission with TransOrder, set IsProcessInSequence=N.
2. Create an order release for every TransOrder either with IsShippable=Y and omit the ReleaseInstruction, or with IsShippable=N and include a ReleaseInstruction.
3. For every TransOrder that Oracle Transportation Management should bulk plan later, set the ProcessingCodeGID to NOPLN.

   If you instead set the ProcessingCodeGID to PLN on each TransOrder in a transmission, Oracle Transportation Management bulk plans these orders on each transmission. Also, Oracle Transportation Management cannot supply a bulk plan history in this case.
4. When Oracle Transportation Management has received all TransOrders to be bulk planned, send a Topic element as the last element in the transmission to start the bulk planning. Set TopicArgName to 'savedQuery' and TopicArgValue to a Query_Name. The saved query must point out the Order Releases you want to include. If you want to supply your own bulk plan ID, in addition, set TopicArgName to 'bulkPlanID' and TopicArgValue to your desired bulk plan ID.

   To be reasonably sure that Oracle Transportation Management has received all your transmissions, allow sufficient amount of time between sending the last TransOrder Transmission and sending the Topic element.

   Note: Oracle Transportation Management plans all order releases that match the saved query, not just the ones within the last transmissions.

   Note: The Topic element must be the last element in the transmission or group. If it is not, Oracle Transportation Management will plan incorrectly.

Transmission Results

When Oracle Transportation Management completes the bulk planning, Oracle Transportation Management sends the results of the bulk plan in a BulkPlan element.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Incrementally Release TransOrder Line from Existing TransOrder

In this scenario, you already have a TransOrder with a large amount of goods in a TransOrderLine in Oracle Transportation Management but now you want to release small amounts of that TransOrderLine with multiple subsequent TransOrders.

To incrementally release TransOrderLines from an order already in Oracle Transportation Management via an integration transmission, do the following:

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation.orderinterface property.

What Data Goes into the Transmission?

1. Make sure the public Order Base - Modify - Incremental Release agent is active.
2. Send a transmission of the record and enter the transaction code U in the TransactionCode element.
3. All your TransOrderLines must be marked IsShippable=N.
4. Always keep IsShippable=N between all these TransOrders.
5. Include a TransOrderHeader/ReleaseInstruction to release a fraction of the amount specified on the original TransOrderLine. If you omit the ReleaseInstruction element, Oracle Transportation Management only saves your order base since you have IsShippable set to N.
6. For each modified TransOrder you send, update the ReleaseInstruction/SequenceNumber and make it unique. If you do not, Oracle Transportation Management keeps the old releases but replaces the content of the release instruction.
7. See the Order Base Manager for a description of the fields.
8. See the TransOrder Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

1. Oracle Transportation Management receives your transmission and starts to process your transmission internally.
2. Oracle Transportation Management starts the public Order Base - Modify - Incremental Release agent.
3. Oracle Transportation Management raises events that in turn can trigger notifications to be sent.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Send TransOrder from Oracle Transportation Management

Required Data

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

In the Order Base Manager.

In the Process Manager, Send Integration Page.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified for the order base.

Transmission Results

Error Messages

Processing Codes

When you send an order to Oracle Transportation Management, you can indicate whether you want Oracle Transportation Management to perform planning functions on it. If Oracle Transportation Management plans orders, it creates shipments from the orders and then executes the shipments as soon as it receives them. If you want to execute orders into shipments at a particular time or after you receive a certain number of orders, do not run the planning function.

Control the details of planning orders in Oracle Transportation Management in the Agent Manager.

In the ProcessingCodeGID element, enter one of the following values:

- **NOPLN**: Instructs Oracle Transportation Management not to plan shipments form the order. This is the default if you omit this element.
- **PLN**: Instructs Oracle Transportation Management to plan shipments from the order release. Oracle Transportation Management will plan multi-stop shipments if appropriate. You must have your TransOrder set up to create an order release for this to work.
- **MSPLN**: Obsolete.

XLane Interface

Use the XLane interface to transmit lane information to Oracle Transportation Management. An XLane connects two geographic points. Associate XLanes with rate records to establish rates between particular locations or areas. The geographic points can be specific, such as a street address, or more general, such as a city, state, country, or region. You can include mileage interface information in the lane transmission.

XLanes provide the basic geographic framework for shipment activity. When you define a lane with specific geographic locations, fewer shipments can use the lane. When you define a lane with less specific geographic information, more shipments qualify for it but processing time on the system may be slowed.

Data Requirements

Before you can transmit lane information to Oracle Transportation Management, you must enter or transmit the location information associated with the lanes.

Billing Interface

Use the Billing interface to send transmissions to an accounting system for customer billing. The billing information represents the amount owed by a customer to the planner of a shipment. The billing transmission includes the customer who is being charged and details of the bill such as the amount due, the date due, and any discount information.

The elements included in the Billing interface transmission are the same as those in the invoice interface. Oracle Transportation Management sends the bill to the involved parties specified on the order.

For consolidated invoices, when you send the Billing interface, all child invoices are also sent.

Oracle Transportation Management sends Billing transmissions to the URL you specify in glog.properties.

Bills can be generated in the Shipment Manager or in the Process Manager.

Voucher Interface

A voucher approves the payment of an invoice. A voucher represents the cost of a shipment owed to a third party such as a service provider. Oracle Transportation Management sends a voucher transmission to a financial system after it has matched a third party charge to a shipment and approved the charge for payment.

Oracle Transportation Management sends voucher transmissions to the URL you specify in glog.properties.

You can create an agent that send the Voucher interface using the agent action Send Voucher Interface. You can also send this interface from the Invoice Manager.

> **Note:** You can send a voucher transmission that cancels or edits a previous voucher.

Oracle Transportation Management determines to whom a payment is due based on the involved parties defined on the order release.

The Shipment element is only included when generating vouchers for parent invoices. When generating a voucher for a child invoice, the Shipment element is not included.

You can optionally use the ShipmentGID element instead of the full Shipment element in order to reduce the size of the Voucher XML.

Process Vouchers

Vouchers are generated in the Process Manager

CSVFileContent Interface

The CSVFileContent element embeds the contents of a CSV file. This element should only be used for setup activities; it is not intended for operational activity.

Insert New CSVFileContent into Oracle Transportation Management

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

What Data Goes into the Transmission?

Each CSVFileContent element can contain only one CSV File. CSVFileContent only supports inserts into the database.

1. Refer to the Data Management Guide document for details on the CSV file format.
2. See the CSVFileContent Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your CSV records in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

DataQuerySummary Interface

Contains the GID of a business object record.

Some external systems may not be prepared for Oracle Transportation Management to send large amounts of data. The DataQuerySummary interface provides a mechanism to send only a summary of the data. The external system can request the individual records from Oracle Transportation Management at appropriate times (e.g. idle times, overnight) by referencing the GID.

Send DataQuery Summary from Oracle Transportation Management

Required Data

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

Mark the Send Summary check box when sending from one of these managers:

- Order Base Manager
- Order Manager (use for order releases)
- Buy Shipment Manager
- Sell Shipment Manager
- Shipment Group Manager
- Bill Manager

- Service Provider Manager
- Location Manager
- Rate Offering Manager
- Rate Record Manager
- Item Manager
- Invoice Manager
- In the Process Manager, Send Integration Page.

What Data Goes into the Transmission?

Oracle Transportation Management includes only the GID of the record.

Transmission Results

Error Messages

FinancialSystemFeed Interface

The FinancialSystemFeed interface allows you to use shipments (either buy or sell) as the final object for managing financials. This alleviates the need to use invoices or bills as the final object for managing financials.

For details on the data requirements and format of this interface, please see the FinancialSystemFeed diagram.

You will use either the SellSideFinancials element or the BuySideFinancials element depending on whether you are using a buy or a sell shipment to manage your financials. All related orders will be passed through based on the allocation information passed through the AllocationBase element.

HazmatGeneric Interface

The HazmatGeneric element specifies a hazardous material. See the Hazardous Materials manager for more information.

> **Note:** To send hazardous material data to Oracle Transportation Management based on specific item information, use the HazmatItem interface.

Insert New Hazmat into Oracle Transportation Management

Required Data

You have to define a technical name for certain hazmats.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

What Data Goes into the Transmission?

1. Set the TransactionCode to I.
2. See the Hazardous Materials Manager online help for a description of the fields.

3. See the HazmatGeneric Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your hazmat definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Modify Hazmat in Oracle Transportation Management

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to U.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the Hazardous Materials Manager online help for a description of the fields.
4. See the HazmatGeneric Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your modified hazmat definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Delete Hazmat in Oracle Transportation Management

Required Data

You must know the HazmatGenericGID of the hazmat you want to delete.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D.
2. Enter the HazmatGenericGID.

Transmission Results

Oracle Transportation Management deletes your hazmat in the database.

Error Messages

If you get a TransmissionReport that reads "CAUGHT THE FOLLOWING EXCEPTION WHILE PROCESSING TRANSACTION:", make sure you are not trying to delete a hazmat that is used elsewhere in Oracle Transportation Management.

If you receive a TransmissionReport, check for integration messages.

HazmatItem Interface

The HazmatItem element specifies an item containing a hazardous material. See the Hazardous Item manager for more information.

> **Note:** To specify a hazardous material, use the HazmatGeneric interface.

Insert New HazmatItem into Oracle Transportation Management

Required Data

You need to define the following before sending a HazmatItem to Oracle Transportation Management:

- STCC ID (49...)
- Package Type
- Hazmat Region
- For which transport modes this item is considered a hazardous item.
- All hazardous materials

In addition, you may need to define the following:

- Hazmat Approval Exemption
- Hazmat Transport Message

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

What Data Goes into the Transmission?

1. Set the TransactionCode to I.
2. See the Hazardous Item Manager online help for a description of the fields.
3. See the HazmatItem Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your hazardous item definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Modify HazmatItem in Oracle Transportation Management

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to U.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the Hazardous Item Manager online help for a description of the fields.
4. See the HazmatItem Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your modified hazardous item definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Delete HazmatItem in Oracle Transportation Management

Required Data

You must know the HazmatItemGID of the hazardous item you want to delete.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D.
2. Enter the HazmatItemGID.

Transmission Results

Oracle Transportation Management deletes your hazardous item definition in the database.

Error Messages

If you get a TransmissionReport that reads "CAUGHT THE FOLLOWING EXCEPTION WHILE PROCESSING TRANSACTION:", make sure you are not trying to delete a hazardous item definition that is used elsewhere in Oracle Transportation Management.

If you receive a TransmissionReport, check for integration messages.

RATE_GEO Interface

Contains a rate record.

RATE_GEO is "database centric" since the Oracle Database auto-generates the schema. The element names directly correspond to the table and column names.

Send RATE_GEO from Oracle Transportation Management

Required Data

For details on what data is required and what is not, as well as the format of the data, please reference the RATE_GEO interface diagram.

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

- In the Rate Record Manager.
- In the Process Manager, Send Integration Page.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified in the rate record.

Insert Rate Records into Oracle Transportation Management

To import rate records, use the CSV utility.

RATE_OFFERING Interface

Contains a rate offering.

RATE_OFFERING is "database centric" since the Oracle Database auto-generates the schema. The element names directly correspond to the table and column names.

Send RATE_GEO from Oracle Transportation Management

Required Data

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

In the Rate Offering Manager.

In the Process Manager, Send Integration Page.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified in the rate offering.

Transmission Results

Error Messages

Insert Rate Records into Oracle Transportation Management

To import rate offerings use the CSV utility.

Schedule Interface

This element is supported both for inbound and outbound, and contains parameters used in the Batch Balance algorithm. The schedule is used as input to build shipments and assign orders into batches. Each order (TransOrder) can be assigned to a Schedule for processing. The ScheduleStatus and Batch elements in the Schedule element are outbound only.

ShipmentGroup Interface

Use this interface to specify shipment group header information and the associated shipments in a group. The ShipmentGroup element is supported for both inbound and outbound messages.

ShipmentGroupTenderOffer

The ShipmentGroupTenderOffer interface is used to notify a service provider of a shipment group pickup. This interface is a wrapper around the ShipmentGroup element, and currently does not accept response back from the service provider.

For details on the elements included in this interface, and which ones are required versus optional, view the ShipmentGroupTenderOffer diagram in the online help system.

ShipmentLink (Related Shipments) Interface

The ShipmentLink interface identifies related shipments and acts as an inbound interface.

With the ShipmentLink interface you can link shipments that you send to Oracle Transportation Management via integration. Use the ShipmentLink interface to identify related shipments using cross-dock and pool de-consolidation locations. The ShipmentLink interface includes the GID for each linked shipment and identifies the stop sequence for the shipments. For instance, you can:

1. Send in info for Shipment 1
2. Send in info for Shipment 2
3. Send in a ShipmentLink linking Shipment 1 and Shipment 2

TransactionAck Interface

This element is inbound only. An external system sends a TransactionAck to Oracle Transportation Management to acknowledge receipt and processing of a transaction.

The TransactionAck contains the original transmission's unique ProcessControlRequestID. This identifies which transaction the TransactionAck acknowledges receipt and processing of.

TransmissionAck Interface

Confirms that Oracle Transportation Management received the transmission.

Send Transmission Ack from Oracle Transportation Management

Required Data

An external system must have sent a transmission to Oracle Transportation Management.

Setup

Your external system must be set up to read response codes to its HTTPPOSTs, otherwise it will not receive a TransmissionAck.

How To Send the TransmissionAck?

Immediately upon receiving a transmission, Oracle Transportation Management automatically echoes a TransmissionAck back to the external system that sent the transmission.

What Data Goes into the TransmissionAck?

The TransmissionAck contains a copy of the original transmission's TransmissionHeader and a reference number (i_transmission_no) assigned to the transmission by Oracle Transportation Management.

Exceptions

The StackTrace element is used when there is an exception with staging the inbound XML.

MessageReport Interface

Lists why the message, sent to Oracle Transportation Management, was not processed successfully.

Send Message Report from Oracle Transportation Management

Required Data

An external system must have sent a message to Oracle Transportation Management.

That message must contain a Message/MessageHeader/AckSpec with a ComMethodGid. The ContactGid can also be specified for the recipient.

Setup

`glog.integration.MessageReport` controls what kind of errors trigger Oracle Transportation Management to send a MessageReport. You can override these settings with the AckSpec element of your message.

How to Send the MessageReport?

Oracle Transportation Management sends the MessageReport after the message has been processed.

Oracle Transportation Management sends the MessageReport via either email or HTTPPost or Queue depending on what the AckSpec element in the MessageHeader of the original message specifies.

What Data Goes into the MessageReport?

The MessageReport consists of a username and password for the receiving system, the message status, processing error code, echoed message header for reference, as well as integration log messages and summary information. It also contains the MessageBody, which includes the TransmissionReport summarizing integration log messages and summary information from the Transmission processing. The MessageReport details all the errors, which must be corrected before the data can be re-transmitted.

BulkContMove Interface

After building a bulk continuous move, Oracle Transportation Management sends this element to an external system. BulkContMove is used to provide statistics about shipments that were selected and linked during a given run of bulk continuous move. This element is supported on the outbound only.

BulkPlan Interface

After finishing a bulk planning session, Oracle Transportation Management sends this element to an external system. BulkPlan contains statistics about the orders that were planned and the shipments that were created.

BulkRating Interface

After finishing a bulk rating session, Oracle Transportation Management sends this element to an external system. BulkRating contains statistics about the orders that were planned and the shipments that were created.

BulkTrailerBuild Interface

After finishing a bulk trailer build, Oracle Transportation Management sends this element to an external system. BulkTrailerBuild contains statistics about the shipment groups that were created during the bulk trailer build process. This element is supported on the outbound only.

GenericStatusUpdate Interface

This element expands on and replaces the following elements:

- LocationStatusUpdate
- TransOrderStatusUpdate
- PaymentStatusUpdate
- OrderReleaseStatusUpdate
- ShipmentStatusUpdate
- VoucherStatusUpdate
- ShipmentGroupStatusUpdate
- ScheduleStatusUpdate

This element updates the external statuses of the corresponding objects. It can update Refnum(s), Remark(s), Status(es), and Indicator(s).

You use the GID and the SequenceNumber elements to identify the object to update. The GID specifies the primary key of the object (e.g. ShipmentGID for the SHIPMENT object). The SequenceNumber, together with the GID element, identifies the object when the GID is insufficient. For example, the S_SHIP_UNIT_LINE object has a S_SHIP_UNIT_LINE_NO field as part of its primary

key, so the SequenceNumber would correspond to the S_SHIP_UNIT_LINE_NO. Other objects requiring the sequence number include the INVOICE_LINEITEM and SHIPMENT_STOP.

When updating voucher reference numbers, you can only update a single voucher reference number per transaction.

The Role of the Transaction Code

The TransactionCode specifies whether the information should be inserted, or updated/replaced. For the Refnum objects that have the qualifier and value as part of the primary key, the TransactionCode indicates whether the new qualifier/value pair should be added (Insert), or used to replace all of the current records with the same qualifier (Update).

For example, the Shipment_Refnum table has a composite primary key made up of the ShipmentGID, RefnumQualifier, and RefnumValue. Assume a shipment has the following ShipmentRefnum Qualifier/Value pairs in the system: CO/A-12345, CO/B-89387, CN/C-83920. If you send a new refnum qualifier/value of CO/D-23849 using the GenericStatusUpdate interface, the TransactionCode would affect the change as follows:

- TransactionCode = I: The new refnum would be added, resulting in all of the following being present in the table: CO/A-12345, CO/B-89387, CN/C-83920, CO/D-23849
- TransactionCode = U: The current reference numbers with the same qualifier would be deleted, and replaced by the new one. In this case, the result would leave the following in the table: CN/C-83920, CO/D-23849

The TransactionCode is only applicable for the Refnum and Remark elements. It is not used for the Status or Indicator elements, which are only intended to be updated.

Topic Interface

This inbound interface allows you to raise a topic and get Oracle Transportation Management to start processing an object. Currently Oracle Transportation Management supports BuildBuySideShipments and BuildSellSideShipments that allows you to start bulk planning. Oracle Transportation Management also supports clearing caches using the interface.

> **Note:** Make sure Oracle Transportation Management has released all your TransOrders before sending the Topic element to Oracle Transportation Management.
>
> **Note:** When including other transactions in the same transmission as the Topic transaction, make the Topic transaction the last in the transmission.

The table lists what each element should contain.

TopicAliasName	TopicArgName	TopicArgValue
BuildBuySideShipments	savedQuery	query_name, e.g. YELLOW_ORDER_REL
BuildSellSideShipments	savedQuery	query_name, e.g. YELLOW_ORDER_REL
glog.server.workflow.adhoc.ClearCaches	cache	partial or full string matching the cache name, e.g. RateOffering

TopicAliasName	TopicArgName	TopicArgValue
	zone	zone name, e.g. Rating or Business
	exactMatch	true, if the cache match should be exact

SShipUnit Interface

The SShipUnit element allows you to identify and update s_ship_unit information on a shipment, without having to identify the individual shipment. SShipUnit supports the use of a defined integration saved query to lookup the s_ship_unit_gid. It also supports updates to multiple s_ship_unit(s) that match the query constraints.

Typically you do not know the Shipment/ShipUnit/ShipUnitGID when a shipment has been built as a result of a TransOrder since Oracle Transportation Management generates the Shipment/ShipUnit/ShipUnitGID arbitrarily. Instead you may know the TransOrder/ShipUnitDetail/ShipUnit/ShipUnitGID. You can define a query to search for the Shipment/ShipUnit/ShipUnitGID that Oracle Transportation Management generated from the original TransOrder.

Alternative Interfaces

For alternatives to using this interface see ActualShipment and TransOrder.

How to Use

To use this element, set:

- IntSavedQueryGID to specify which query you want to use. If the query you specify here does not return any results, Oracle Transportation Management generates an error message and stops processing the SShipUnit. If you omit the IntSavedQuery element altogether, Oracle Transportation Management just inserts or updates (depending on your TransactionCode) the information in your SShipUnit element. Oracle Transportation Management contains two predefined queries that you can change.
- IntSavedQueryArg to arguments that can be referred to in the queries. For example, TRACKING_TAG1=A123456789. If you omit this element, your IntSavedQueryGID must point to a query that uses XPath instead.
- IsMultiMatch to Y or N to allow multiple ship unit records to be returned from the query or not. If you specify N and your query returns multiple ship unit records, Oracle Transportation Management generates an error message.
- TransactionCode to I, IU, U, D, or RC. You can only delete a ship unit if it is not referenced by for example a shipment.
- ReplaceChildren/ManagedChild to what part or parts of the ShipUnit element you want to update. If you omit this element and specify TransactionCode = RC, Oracle Transportation Management replaces all information in the existing ShipUnit(s) with the information in your inbound SShipUnit/ShipUnit.

Then you can enter your updated information in the ShipUnit element.

After receiving an SShipUnit transmission, Oracle Transportation Management searches for the ShipUnitGID (Oracle Transportation Management ignores the ShipUnitGID in your ShipUnit element).

Based on the TransactionCode, Oracle Transportation Management then updates the ShipUnit(s) that it finds.

TransOrderStatus Interface

Service providers and other third parties transmit order base event information to Oracle Transportation Management with the TransOrderStatus interface.

Insert New TransOrder Status into Oracle Transportation Management

Required Data

Before you can send TransOrderStatus transmissions to Oracle Transportation Management, you must set up the following:

- User accounts for service providers in Oracle Transportation Management.
- Order Base Status Codes
- Order Base Status Reason Codes
- Order Base Event Groups
- Order Base Reason Groups
- Corporations for service providers

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

What Data Goes into the Transmission?

1. To ensure that Oracle Transportation Management processes multiple TransOrderStatus transactions in the order you intend, set IsProcessInSequence to Y in the TransmissionHeader.
2. Identify which object the status applies to. Set StatusLevel, OrderRefnum and ServiceProviderAlias.

   If you try to match your TransOrderStatus against a reference number and Oracle Transportation Management finds multiple matches for the reference number you supplied, Oracle Transportation Management gives you an error.
3. Include the time when the event occurred. Your best option is to include the TimeZoneGID. If you cannot do this, set the TimeZoneGID to Local. In this case, Oracle Transportation Management saves and displays the event date as entered, ignoring user preferences.
4. Enter your status information.
5. See the Order Base Event Manager online help for a description of the fields.
6. See the TransOrderStatus Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your information in the database.

Agents might have been set up to act upon receiving certain statuses.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Contact Interface

The Contact element specifies a contact. A location may have multiple contacts.

Insert New Contact into Oracle Transportation Management

Required Data

If you want to associate your contact with an Oracle Transportation Management user ID, location, or external system you must have defined these first.

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

How To Send the Transmission?

A contact may be transmitted to Oracle Transportation Management as a stand-alone transmission, or embedded in other transmissions. Sending a contact embedded in another transmission can save time.

What Data Goes into the Transmission?

1. Set the TransactionCode to I.
2. See the Contact Manager online help for a description of the fields.
3. See the Contact Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your contact definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Modify Contact

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation.locationinterface property.

What Data Goes into the Transmission?

1. Set the TransactionCode to U.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the Contact Manager online help for a description of the fields.

4. See the Contact Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your modified contact definition in the database.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Delete Contact

Required Data

You must know the ContactGID of the location you want to delete.

Setup

You control validation of incoming transmissions with the glog.integration.validation.locationinterface property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D.
2. Enter the ContactGID.

Transmission Results

Oracle Transportation Management deletes your contact in the database.

Error Messages

If you get a TransmissionReport that reads "CAUGHT THE FOLLOWING EXCEPTION WHILE PROCESSING TRANSACTION:", make sure you are not trying to delete a contact that is used elsewhere in Oracle Transportation Management.

If you receive a TransmissionReport, check for integration messages.

ContactGroup Interface

The ContactGroup Interface represents a list of contacts used for notification. The elements of the interface mirror the fields found in the Contact Group Manager.

A contact group is a collection of contacts that can act as a broadcast mechanism to a set of email, fax, or HTTP addresses using common language, communication method, and preferences. Or it can be used to notify several people at one time about an event that has occurred with contact-specific language, communication, and preference settings.

SKU Interface

SKU defines a stock keeping unit including what quantities to keep in stock, and the actual amount in the warehouse.

Insert New

Required Data

You need to have the following defined in Oracle Transportation Management:

- PackagedItem
- Location of the warehouse
- Supplier and owner corporation of the SKU
- ShipUnit holding the SKU
- Contact information for the involved party

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to I.
2. See the SKU Manager online help for a description of the fields.
3. See the SKU Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your SKU information in the database.

You can set up involved parties to be notified when certain changes (events) occur.

You can also set up an agent to monitor the quantity on hand and notify involved parties when necessary.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Updates throughout the Day

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to IU.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the SKU Manager online help for a description of the fields.
4. See the SKU Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your modified information in the database.

You can set up involved parties to be notified when certain changes (events) occur.

You can also set up an agent to monitor the quantity on hand and notify involved parties when necessary.

> **Note:** This will NOT delete any child records from the sku_descriptor table. See the next section if you want to completely replace an individual SKU.

Error Messages

If you receive a TransmissionReport, check for integration messages.

Replace an Individual SKU

Required Data

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

1. Set the TransactionCode to D for the first transaction and I for the second.
2. Enter data in the elements that need to be updated and in the required elements.
3. See the SKU Manager online help for a description of the fields.
4. See the SKU Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management deletes you existing SKU in the database and then insert your new information.

You can set up involved parties to be notified when certain changes (events) occur.

You can also set up an agent to monitor the quantity on hand and notify involved parties when necessary.

Error Messages

If you receive a TransmissionReport, check for integration messages.

How To Structure Your Data

If you have large amounts of highly complex, nested inventory information, you should use the XML column in the SKU_DESCRIPTOR table to store that information, rather than using nested SKU_DESCRIPTOR records. However, it will not be possible to search for SKUs using the XML info. You will only be able to search on the non-XML columns in the SKU and SKU_DESCRIPTOR tables.

If you have small or medium amounts of less complex inventory information, you can use nested SKU_DESCRIPTOR records instead. Using this method, it will be possible to find a SKU by a sub-descriptor.

SKU Table

The following sample SKU record shows a part used to make Novelty phones. The Novelty stock code is 2002, which is used to form the XID. This corresponds to the packaged_item novelty.8946. The warehouse is novelty.wh1. The supplier is General Electric, who also currently owns the inventory.

```
SKU_GID = novelty.2002-wh1
SKU_xid = 2002-wh1
Packaged_item_GID = novelty.8946
Warehouse_location_GID = novelty.wh1
Supplier_corporation_GID = novelty.ge
Owner_corporation_GID = novelty.ge
Quantity_on_hand = 1800
Min_level = 100
Max_level = 2000
Domain_name = novelty
```

SKU_DESCRIPTOR table - BLOB

Oracle Transportation Management cannot show BLOBs in tree view of the inventory manager.

This section illustrates the relational approach to storing SKU descriptor and sub-descriptor information in using the SKU_DESCRIPTOR table. This method would be used when it is necessary to use standard SQL to search SKU descriptor data.

The example below shows a top-level SKU_DESCRIPTOR record. Notice that the parent_sku_descriptor_seq is null.

```
SKU_GID = novelty.2002-wh1
SKU_descriptor_seq = 1
SKU_descriptor_type = status
SKU_desriptor_value = held
SKU_descriptor_quantity = 1000
Parent_sku_descriptor_seq = null
Domain_name = novelty
```

The example below shows a level-2 SKU_DESCRIPTOR record. The parent_sku_descriptor_seq is set to 1, pointing to the previous example.

```
SKU_GID = novelty.2002-wh1
SKU_descriptor_seq = 2
SKU_descriptor_type = reason
SKU_descriptor_value = damaged
SKU_descriptor_quantity = 600
Parent_sku_descriptor_seq = 1
Domain_name = novelty
```

The example below shows a level-3 SKU_DESCRIPTOR record. The parent_sku_descriptor_seq is set to 2, pointing to the previous example.

```
SKU_GID = novelty.2002-wh1
SKU_descriptor_seq = 3
SKU_descriptor_type = batch
SKU_descriptor_value = 001
SKU_descriptor_quantity = 250
Parent_sku_descriptor_seq = 2
Domain_name = novelty
```



```
SKU_GID = novelty.2002
SKU_descriptor_seq = 4
SKU_descriptor_type = batch
SKU_descriptor_value = 002
SKU_descriptor_quantity = 300
Parent_sku_descriptor_seq = 2
Domain_name = novelty

SKU_GID = novelty.2002
SKU_descriptor_seq = 5
SKU_descriptor_type = batch
SKU_descriptor_value = 003
SKU_descriptor_quantity = 50
Parent_sku_descriptor_seq = 2
Domain_name = novelty
```

SKU_DESCRIPTOR Table - XML

If the SKU descriptor information need not be fully searchable using standard SQL, then the XML column in the SKU_DESCRIPTOR table may be used to represent the information at level 2 and below. In other words, it would be possible to use standard SQL to search for a SKU descriptor, but not for a SKU sub-descriptor.

An example situation where the XML method may not be appropriate would be where the top level SKU is a combination of shoes of different styles. The top level SKU_DESCRIPTOR records would have one row for each style. The level 2 SKU_DESCRIPTOR would have counts of sizes within each style. A query to determine the total inventory of size 9 shoes across all styles would not be possible using the XML method. You can think of similar examples for the auto industry, i.e. find the inventory of all cars with anti-lock brakes, etc.

When using the XML method for representing detailed SKU_DESCRIPTOR information, each client implementation will be responsible for developing their own industry-specific XML schema for that information. By default, the UI will display this information in a nicely formatted manner. The UI provides a mechanism whereby you can install custom XSL for formatting information. However, this XSL file is purely optional.

Below is a snippet of how the information from the previous section might appear in the database if the XML approach is used instead of the nested SKU_DESCRIPTOR method. In this case, the parent_sku_descriptor_seq column is always null, and the XML column is used instead. In this case, the top level status information is available relationally. However, the lower level descriptors within that status are represented inside the XML.

```
SKU_GID=novelty.2002-wh1
SKU_descriptor_seq = 1
SKU_descriptor_type = status
SKU_descriptor_value = held
SKU_descriptor_quantity = 1000
Domain_name = novelty
Xml =
<SkuDescriptor>
  <type>damaged</type>
  <value>001</value>
  <quantity>600</quantity>
<SkuDescriptor>
  <type>batch</type>
  <value>001</value>
  <quantity>250</quantity>
```



```
</SkuDescriptor>
<SkuDescriptor>
  <type>batch</type>
  <value>002</value>
  <quantity>300</quantity>
</SkuDescriptor>
… etc …
</SkuDescriptor>
```

SkuTransaction Interface

SkuTransaction represents a shipment of SKUs arriving or leaving the warehouse. This is separate from shipments created in Oracle Transportation Management. Also, SkuTransactions do not update the SKU table.

Ship SKU To or From the Warehouse

Required Data

You need to have the following defined in Oracle Transportation Management:

- SKU
- PackagedItem
- Location of the warehouse
- supplier and owner corporation of the SKU

Setup

You control validation of incoming transmissions with the glog.integration.validation property.

What Data Goes into the Transmission?

You can only insert SkuTransactions, so there is no TransactionCode.

1. See the SKU Manager online help for a description of the fields.
2. See the SkuTransaction Diagram in the XML schema to learn which elements are required. To view the diagrams use an XML application like XML Spy.

Transmission Results

Oracle Transportation Management saves your information in the database.

You can set up involved parties to be notified when certain changes (events) occur.

You can also set up an agent to monitor the quantity on hand and notify involved parties when necessary.

Error Messages

If you receive a TransmissionReport, check for integration messages.

OBLine Interface

This is used to send out the order base line information.

Send OBLine from Oracle Transportation Management

Required Data

You must have created the order base line.

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

In the Order Base Line Manager.

With an agent. See the agent action called Send Integration.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified for the order base line.

OrderMovementReplace Interface

Some itineraries contain pre-transport legs, ocean legs, and customer delivery legs. When an order is scheduled onto an itinerary, it will be split into component order movements corresponding to each leg it is routed on. Production lot schedule information will be used to further split the first leg order movement into multiple order movements, all corresponding to the same leg and indicating how the order is available for movement on that leg. Similarly, the delivery line information will be used to split the delivery transport leg into multiple order movements all corresponding to that leg.

The OrderMovementReplace (OMR) interface is used to bring in the production lot and delivery line information and use it to carry out the necessary modifications to the order movements on the corresponding legs. This interface is only supported on the inbound.

The OrderMovementReplace interface has the following elements/attributes:

Name	Description
OMR Name	Name of this set of information.
Process Time	Indicates the time at which the information contained is current.
OMR Type	Indicates whether it is supply information or demand information.
Time Interval Type	Indicates whether the span between the early and late times represents a times window or availability or of continuous flow. Available values are "window" and "flow."
Description	For informational use only.
Order Release ID	The order release id to which order movements are related.
Order Ship Unit ID	The field is optional.

Name	Description
Count	The quantity of the ship unit.
Early Time	Early available time.
Late Time	Late available time.
Time Series Type	Indicates whether this information is about the past or future. Available values are "future" and "history."
Unit Weight	The weight of each unit in the ship unit.
Unit Volume	The volume of each unit in the ship unit.
Unit Length	The length of each unit in the ship unit.
Unit Height	The height of each unit in the ship unit.
Unit Width	The width of each unit in the ship unit.
Unit Diameter	The diameter of each unit in the ship unit.

OBShipUnit Interface

This is used to send out order base ship unit information.

Send OB Ship Unit from Oracle Transportation Management

Required Data

You must have created the order base ship unit.

Setup

You must have created the external system you want to send to.

How To Send the Transmission?

In the Order Base Ship Unit Manager.

With an agent. See the agent action called Send Integration.

What Data Goes into the Transmission?

Oracle Transportation Management includes all data specified for the order base ship unit.

Transmission Results

Error Messages

Voyage Interface

The Voyage interface is used to send world-wide vessel schedule information to external systems. This interface is supported on the outbound only.

The elements of this interface mirror the fields found in the Voyage Manager. Voyage schedules define the rotation of a carrier's vessel as the vessel goes from a set of loading points (departure ports) to a set of unloading ports (arrival ports).

Since this element is supported on the outbound only, a common way to add voyage data is via the Voyage Manager in the user interface (good for smaller updates), or via the Voyage Data API (good for larger updates), which is described in detail in the Data Management Guide.

BookingLineAmendment Interface

The BookingLineAmendment interface is used to send booking line changes out of the system. This interface is supported on the outbound only.

A BookingLineAmendment can be initiated when an order or a portion of an order is added or removed from a shipment, thereby changing the charter voyage to which it was assigned. The BookingLineAmendment will contain either the BookLineAmendViaRelease or the BookLineAmendViaServiceProvider, but not both. The selection is based on the agent used to send the notification for the message.

CharterVoyage Interface

The CharterVoyage interface is used to specify the charter voyage for creating a consol shipment. It is supported on both the inbound and the outbound.

A charter voyage represents an ocean transport movement by a carrier from a loading port to a discharge port. Within a charter voyage, there are several Stowage Modes, which represent, at a conceptual level, separate “compartments” within the charter voyage. There is also capacity associated with each stowage mode as defined on a consol that you create for each stowage mode defined on the charter voyage. This capacity controls the orders that can be booked on the charter voyage.

For a charter voyage, and each of its defined stowage modes, you can create a consol that has a single empty shipment attached. For each voyage, one consol is automatically created for each stowage mode defined on the voyage. A shipment is also created for each consol at the same time.

Most of the elements included in the CharterVoyage interface follow the fields available in the Charter Voyage manager and the Charter Voyage Stowage Details.

Consol Interface

The Consol interface is used to specify the shipment consolidator. It is supported on both the inbound and the outbound. A consol can be created for a charter voyage or air schedule (flight).

A charter voyage consol represents the weight, volume, FEU/TEU capacities of a specific stowage mode on a specific charter voyage. It captures the allocated, maximum, committed, booked, and

produced capacity values when the status of a consol is changed as a result of booking orders on a shipment that is related to the consol.

For a freight forwarder, the consol is considered a group of house bills or a set of sell shipments. All actions related to manipulating a consol should be performed from the perspective of a sell shipment. For example, adding freight to a consol would be performed by selecting sell shipments to add to consol.

For example, a freight forwarder starts with a group of house bills or a set of sell shipments. They have also reserved flights. For each flight reservation, there is a consol for defining the reserved capacity of the flight. The sell shipments are then booked to consols to create buy shipments.

Claim Interface

The Claim interface is used to specify damage claims. This interface is supported on both the inbound and the outbound.

A claim contains information for Oracle Transportation Management or non-Oracle Transportation Management damaged shipments, can be used to notify parties of their involvement with a claim, and can tracks status changes that occur throughout the claim process. A claim consists of any freight damage that occurs to a shipment prior to taking ownership/responsibility of the shipment. Claims in Oracle Transportation Management are not dependent on pre-existing Oracle Transportation Management orders or shipments. Oracle Transportation Management can create a claim about a non-related Oracle Transportation Management order as well as pull data from existing Oracle Transportation Management shipments and orders.

Document Interface

The Document interface provides a consistent way to send and receive business documents in and out of the system. It is supported on both the inbound and the outbound.

Business documents are objects that contain the contents of a traditional document, like a bill of lading for example, in electronic format. This enables you to send and receive the business documents via integration.

Some of the data included in the Document interface includes information relating to a document's owner, its content, its parameters, and any involved parties.

SkuEvent Interface

The SkuEvent interface specifies a SKU event, which describes activities on SKUs. This enters order events to inform planners, shippers, order owners, and other involved parties about actions related to an SKU.

TransOrderLink Interface

The TransOrderLink interface is used to establish a link between order base objects. This provides the ability to maintain orders in various states along with their relationships. Within the interface, the PrevObjectGID element and the NextObjectGID element refer to TransOrderGID(s). This interface is supported on the inbound only.

RouteTemplate Interface

RouteTemplate represents the plan for a cooperative route. A cooperative route is a linking of lanes that have been identified to have sufficient recurring volume of shipments to form a good route for a fleet or dedicated vehicle.

Quote Interface

The quote interface allows customer service representatives, CSR, to supply their customers with transportation quotes. It is supported on the outbound.

General quoting process:

- The CSR receives a request for quote.
- CSR will enter necessary customer/shipping information within the system.
- CSR will determine appropriate rate(s) (enter from within the system or external to the system).
- Rate information gets applied to the quote.
- Quote is saved within the system and sent out to involved parties.
- Quote is monitored and/or follow up is performed.
- Customer accepts/rejects quote.
- If accepted, quote is converted to order.

Driver Interface

Driver interface is used to specify driver data to Oracle Transportation Management. The interface is supported both inbound and outbound. Some of the key attributes of a driver are: Driver GID, First Name, Default Home Location, Contact, Rate offering, Status, and Remarks. Apart from these, the other key sub-entities defined for a Driver are Driver Team data, Driver Special Services, Driver CDL information, Driver Types, and Driver Calendar.

PowerUnit Interface

PowerUnit interface is used to specify power unit (such as a tractor) data to Oracle Transportation Management through Asset Management Systems or any other systems. The interface is supported both inbound and outbound. Some of the key attributes of a power unit include: Power Unit Name, Power Unit Type, Special Services, and Remarks.

DriverCalendarEvent Interface

Specifies a Driver Calendar event. This describes calendar event information on the driver. This interface is supported inbound.

WorkInvoice Interface

WorkInvoice specifies a record of driver activities for a shipment. This interface is supported inbound and outbound but it is specifically used to send a record to payroll outbound from Oracle Transportation Management which can be used to calculate driver pay. It also includes work invoice activities which defines list of activities performed by the driver for the shipment. A single work invoice is generated for a driver team by Oracle Transportation Management. The Driver interface inside the Work Invoice should include a Driver interface for the secondary driver.

Equipment Interface

Equipment Interface represents the equipment assigned to a shipment. This interface is supported both inbound and outbound. Some of the key attributes are Equipment Group GID, Equipment Type GID, Equipment Special Services, Remark, and Status.

CSVDataLoad Interface

CSVDataLoad provides the capability to embed the contents of a CSV file for insertion into the database. Each CSVDataLoad element can contain only one CSV File. This interface should only be used for small sets of data.

User Interface

User interface is an interface used for representing an Oracle Transportation Management user. This interface is supported inbound to load Oracle Transportation Management user data.

ActivityTimeDef Interface

The interface is used to specify fields for allowing multiple fixed and variable stop times for each location role based on transport handling unit and commodity. It is supported both inbound and outbound.

DemurrageTransaction Interface

Demurrage is the term used for the fees that are charged by a carrier to a customer for the use of assets beyond a contracted free time for loading and unloading. The concept of a daily charge or "per diem" is an industry standard method to share the cost of ownership or equipment while being used by a customer or carrier other than the owner of the equipment.

It is supported for outbound only.

OrderMovement Interface

Order movement interface is used to specify part of the routing of an order. It is also used to specify a collection of ship units that are unplanned for certain part of a route. The interface is supported both inbound and outbound.

GtmItem Interface

The GtmItem interface provides a way to transmit item and classification information to Oracle Transportation Management. Items represent the freight being shipped. The ItemMaster includes packaging elements, elements that describe the item, and any NMFC, STCC, SITC, or HTS codes that apply. In addition, a general ledger GID or accessorial charges can be included. The GtmItemClassification includes product classification type and code.

GtmParty Interface

Global Trade Management holds a wide spectrum of trade paryt information in its database. The GtmParty interface provides inbound and outbound interfaces to exchange party information with outside sources. The partyScreeningResult element is for outbound only.

GtmTransaction Interface

The GtmTransaction interface allows users to bring in shipment or order release information for screening purpose.

GtmRegistration Interface

The GtmRegistration interface will allow users to upload the information of the registrations existing in external systems that are relevant for trade purposes. GTM supports different type of registrations: for parties (e.g.: legal entities, company branches, customers, carriers, etc.) and for items. Companies could also create the registrations in GTM and pass them to external systems that will require them later.

GtmShipment Interface

The GtmShipment (Customs Shipment) interface will allow the users to upload shipments from external systems that are relevant to trade. Examples could be: import shipment, export shipment, inter-state shipment. The interface can be also used to pass the shipments information including charges and relevant information to an external system such as an ERP Inventory Receiving or Landed Cost Management application.

8. Integration Messages

This chapter lists integration messages, describes why the message occurs, and describes what you need to do as a result of receiving the message.

You might find these error messages in a TransmissionReport element. (ILogMan.java).

Heading	Data
Message:	public final static String Invalid Date Format Text = "THE DATE ELEMENT {0} WITH VALUE {1} IS NOT OF FORMAT YYYYMMDDHHMMSS";
Occurs When:	An invalid date format error occurs when the date is not provided in the format YYYYMMDDHHMMSS.
Corrective Action:	Enter the date using the required format.

Heading	Data
Message:	public final static String dataConversionErrorText = "DATA CONVERSION FAILED FOR THE ELEMENT {0} WITH VALUE {1}";
Occurs When:	A data conversion error occurs when character data cannot be converted to an internal data type.
Corrective Action:	Eliminate extraneous characters from the data element.

Heading	Data
Message:	public final static String duplicateKeyErrorText = "THE ELEMENT(S) {0} WITH VALUE(S) {1} IS A DUPLICATE PRIMARY KEY";
Occurs When:	A duplicate Key error occurs when the primary key for a given element already exists in the G-Log database.
Corrective Action:	Change the Transaction Code element from I to IU.

Heading	Data
Message:	public final static String PKNotFoundErrorText = "THE PRIMARY KEY ELEMENT(S) {0} WITH VALUE(S) {1} COULD NOT BE FOUND IN TABLE {2} COLUMN(S) {3}";

Heading	Data
Occurs When:	A PKNotFoundError most often occurs when an attempt is made to update or delete data that does not exist in the G-Log database.
Corrective Action:	If your Transaction Code is U, then use UI instead. If your Transaction Code is D, then there is no corrective action.

Heading	Data
Message:	public final static String FKNotFoundErrorText = "THE FOREIGN KEY ELEMENT {0} WITH VALUE {1} COULD NOT BE FOUND IN TABLE {2} COLUMN {3}";
Occurs When:	A FK Not Found Error occurs when a referenced primary key does not exist in the GLog database.
Corrective Action:	Correct the XML data value such that it refers to a primary key that does exist in the G-Log database.

Heading	Data
Message:	public final static String missing RequiredElementErrorText = "THE REQUIRED ELEMENT {0} IS MISSING";
Occurs When:	A missing required element error occurs when a required element has been omitted from a GLogXML element.
Corrective Action:	Provide the missing required element in your XML data.

Heading	Data
Message:	public final static String ITransactionNoNotFoundErrorText = "THE I_TRANSACTION_NO WITH VALUE {0} DOESN'T EXIST IN THE DATABASE";
Occurs When:	An ITransactionNoNotFoundError occurs when a GLogXMLElement, such as a Tender Response, refers to a transaction number that does not exist in the G-Log I_TRANSACTION table.
Corrective Action:	Refer to an existing transaction number.

Heading	Data

Heading	Data
Message:	public final static String invalidTransactionCodeErrorText = "THE TRANSACTION CODE {0} is not valid. Valid codes are I,U,D,IU,UI,NP";
Occurs When:	An invalid transaction code error occurs when the Transaction Code element is specified with an invalid value.
Corrective Action:	Specify a valid Transaction Code in your XML data.

Heading	Data
Message:	public final static String transactionCodeNotSupportedText = "THE TRANSACTION CODE {0} IS NOT SUPPORTED IN THIS INTERFACE. VALID CODES ARE {1}";
Occurs When:	A transactionCodeNotSupported occurs when the TransactionCode element is specified with an unsupported value.
Corrective Action:	Correct the XML data to specify a valid TransactionCode.

Heading	Data
Message:	public final static String conflictingElementErrorText = "THE ELEMENT {0} AND THE ELEMENT {1} CANNOT BOTH BE SPECIFIED";
Occurs When:	A conflicting element error occurs when two elements have been provided in a G-Log XML Element, when only one out of the two may be used.
Corrective Action:	Eliminate one of the two conflicting elements in your XML data.

Heading	Data
Message:	public final static String invalidNumberFormatErrorText = "THE NUMBER ELEMENT {0} WITH VALUE {1} IS NOT IN A NUMBER FORMAT";
Occurs When:	An invalid number format error occurs when non-numeric characters are specified in a numeric element.
Corrective Action:	Eliminate the non-numeric characters in your XML data.

Heading	Data
Message:	public final static String missingElementErrorText = "WE REQUIRE A {0} ELEMENT WITH VALUE {1} IN THE {2} ELEMENT";
Occurs When:	A missing element error occurs when an element with a particular value is required.
Corrective Action:	Provide the element with the required value as indicated in the error message.

Heading	Data
Message:	public final static String invalidBooleanErrorText = "THE BOOLEAN ELEMENT {0} WITH VALUE {1} MUST BE EITHER Y or N";
Occurs When:	An invalid Boolean error occurs when a Boolean element is provided with a value other than Y or N.
Corrective Action:	Provide either Y or N in the indicated Boolean element in your XML data.

Heading	Data
Message:	public final static String invalidActionCodeErrorText = "THE ACTION CODE {0} is not valid. Valid codes are A, D";
Occurs When:	An invalid action code error occurs when an action code is specified with a value other than A or D.
Corrective Action:	Provide either A or D in your XML data.

Heading	Data
Message:	public final static String invalidCodeErrorText = "THE ELEMENT {0} WITH VALUE {1} is not valid. Valid codes are {2}";
Occurs When:	An invalidCodeError occurs when a code is specified that is not valid for that element
Corrective Action:	Correct the XML data to provide a valid value.

Heading	Data

Heading	Data
Message:	public final static String invalidActivityErrorText = "THE ACTIVITY {0} is not valid. Valid codes are D, P or O";
Occurs When:	An invalidActivityError occurs when an activity is specified with a value other than P, D or O.
Corrective Action:	Correct the XML data to provide anyone of D, P or O.

Heading	Data
Message:	public final static String transactionProcessorExceptionText = "CAUGHT THE FOLLOWING EXCEPTION WHILE PROCESSING TRANSACTION: {0}";
Occurs When:	A transactionProcessorException occurs when the integration layer validation method has not checked for a given error condition, and the condition is caught by the underlying database. Most often this error occurs when an attempt is made to update the Oracle Transportation Management database so that one or more database referential integrity constraints are violated. For example, this error occurs if you attempt to delete a transportation order after one or more releases have been created.
Corrective Action:	No particular correction can be defined. Analyze the error based on the requirements of the data in the underlying database.

Heading	Data
Message:	public final static String maxLengthExceededErrorText = "ELEMENT {0} VALUE {1} HAS LENGTH {2} WHICH EXCEEDS THE MAX LENGTH {3} OF TABLE {4} COLUMN {5}";
Occurs When:	The length of an element value exceeds the length of the corresponding database column
Corrective Action:	Correct the XML data to provide a value which does not exceed the maximum length.

Heading	Data
Message:	public final static String maxLengthExceededErrorText3 = "ELEMENT {0} VALUE {1} EXCEEDS THE MAX LENGTH {2}";

Heading	Data
Occurs When:	The length of an element value exceeds the length of the corresponding database column
Corrective Action:	Correct the XML data to provide a value which does not exceed the maximum length.

Heading	Data
Message:	public final static String validateMaxLengthErrorText = "ERROR VALIDATING MAX LENGTH OF ELEMENT {0} VALUE {1} - CHECK TABLE NAME {2} COLUMN NAME {3}";
Occurs When:	The specified table/column information could not be found.
Corrective Action:	Call Support.

Heading	Data
Message:	public final static String transactionSuccessText = "TRANSACTION NUMBER {0} APPLICATION {1} PRIMARY KEY {2} TRANSACTION CODE {3} SUCCESSFULLY PROCESSED";
Occurs When:	A transaction success informational message occurs when a transaction has been successfully processed.
Corrective Action:	No action needed.

Heading	Data
Message:	public final static String matchMultipleShipmentErrorText = "UNABLE TO PROCESS DUE TO MULTIPLE SHIPMENTS MATCHED ON {0}";
Occurs When:	A matchMultipleShipment error message occurs when ShipmentRefnums/EquipmentNumber match different Shipment_GID. This type of error happens when receiving a TenderResponse or a ShipmentStatus.
Corrective Action:	Correct the XML data to provide ShipmentRefnum values which correspond to the same Shipment.

Heading	Data

Heading	Data
Message:	public final static String missingRequiredDataErrorText = "WE REQUIRE A {0} ELEMENT WITH A VALUE MATCHING A {1} IN THE {2} TABLE";
Occurs When:	A missingRequiredDataError occurs when an element with a particular value is missing from the database table.
Corrective Action:	Correct the database data to provide the value as indicated in the error message.

Heading	Data
Message:	public final static String orderNotModifiableText = "ORDER {0} IS NOT MODIFIABLE AND SO COULD NOT BE MODIFIED OR DELETED.";
Occurs When:	An orderNotModifiable informational message occurs when the status on an order is not "WKFLW_ORDER_OB_MODIFIABLE". This can occur if an order is in a state that restricts it from being modified, or an agent is setup to restrict modification.
Corrective Action:	If you want to be able to modify the order, you may have to change the state of the order or modify the agent that handles order modifications.

Heading	Data
Message:	public final static String reDoTransmissionErrorText = "UNABLE TO raiseNewXMLTopicsForRedoTransmissions, STACK TRACE: {0}";
Occurs When:	
Corrective Action:	None.

Heading	Data
Message:	public final static String savedQueryNoDataFoundErrorText = "THE SAVED QUERY {0} RETURNS NO DATA";
Occurs When:	The saved query in the SShipUnit element did not return any values.
Corrective Action:	Verify that the integration saved queries are correct and that the desired shipunits exist.

9. Oracle Advanced Queuing

Oracle Advanced Queuing (OAQ) provides an alternate way of sending and receiving XML transmissions to/from Oracle Transportation Management. The main benefit to using OAQ is the added level of guaranteed message delivery provided by a persistent message queue.

To use the OAQ functionality in Oracle Transportation Management, the following setup steps must be performed.

Step 1 –Create Queue Table(s)

The default implementation for OAQ in Oracle Transportation Management relies on a database table called INTG_QUEUE. The table is a point-to-point (single consumer) queue table. The table should be available with the installation of the Oracle Transportation Management database.

The OAQ functionality is not restricted to a single queue table. Additional queue tables can be created as needed, although a single queue table can also support multiple queues (see Step 2 below). The following procedure is available to create additional queue tables.

```
procedure create_int_queue_table(p_table_name varchar2,
   p_comment varchar2,
   p_table_space varchar2 default 'data',
   p_multiple_consumers  boolean Default false );
```

The procedure supports creating multi-consumer queue tables using the p_multiple_consumers argument. The only requirement for creating a queue table is inbound queues that Oracle Transportation Management will read from must be created as a point-to-point (single consumer) table.

Example:

```
Sqlplus> execute
pkg_queue_management.create_int_queue_table('queue_test_table',  'This is for
test only');
```

Alternatively, for the multi-consumer:

```
Sqlplus> execute
pkg_queue_management.create_int_queue_table('queue_test_table',  'This is for
test only', 'data', true);
```

The queue tables created use a custom data type called INTG_QUEUE_MESSAGE. The custom data type supports additional fields used for communication. The definition of the INTG_QUEUE_MESSAGE is:

ID	Type	Description
refnum	varchar2(101)	Can be assigned by client system for message referencing.
subject	varchar2(500)	Arbitrary text field definable by the client.
transmission_no	number	Oracle Transportation Management assigned transmission number.

ID	Type	Description
external_system_id	varchar2(101)	Overrides the external system GID in the TransmissionHeader.
user_name	varchar2(128)	Used for user authentication instead of specifying in the TransmissionHeader in the XML.
password	varchar2(128)	Used for user authentication instead of specifying in the TransmissionHeader in the XML.
Xml	clob	Contains the XML transmission.

Step 2 – Setup Required Inbound Queues

For inbound processing of the XML, a set of four queues are required. The queues are:

- Inbound Queue (originally defined as inbound_aq in release 4.0)
- XML Topic Queue (originally defined as xml_stage_aq in release 4.0)
- Ack Queue (originally defined as ack_aq in release 4.0)
- Exception Queue (originally defined as exception_aq in release 4.0)

A query_replay_aq is also needed for responding to Remote Query transactions such as Rate Inquiry (RIQ).

The following diagram shows the communication from the client to the database, as well as a high level depiction of the processing in the database.

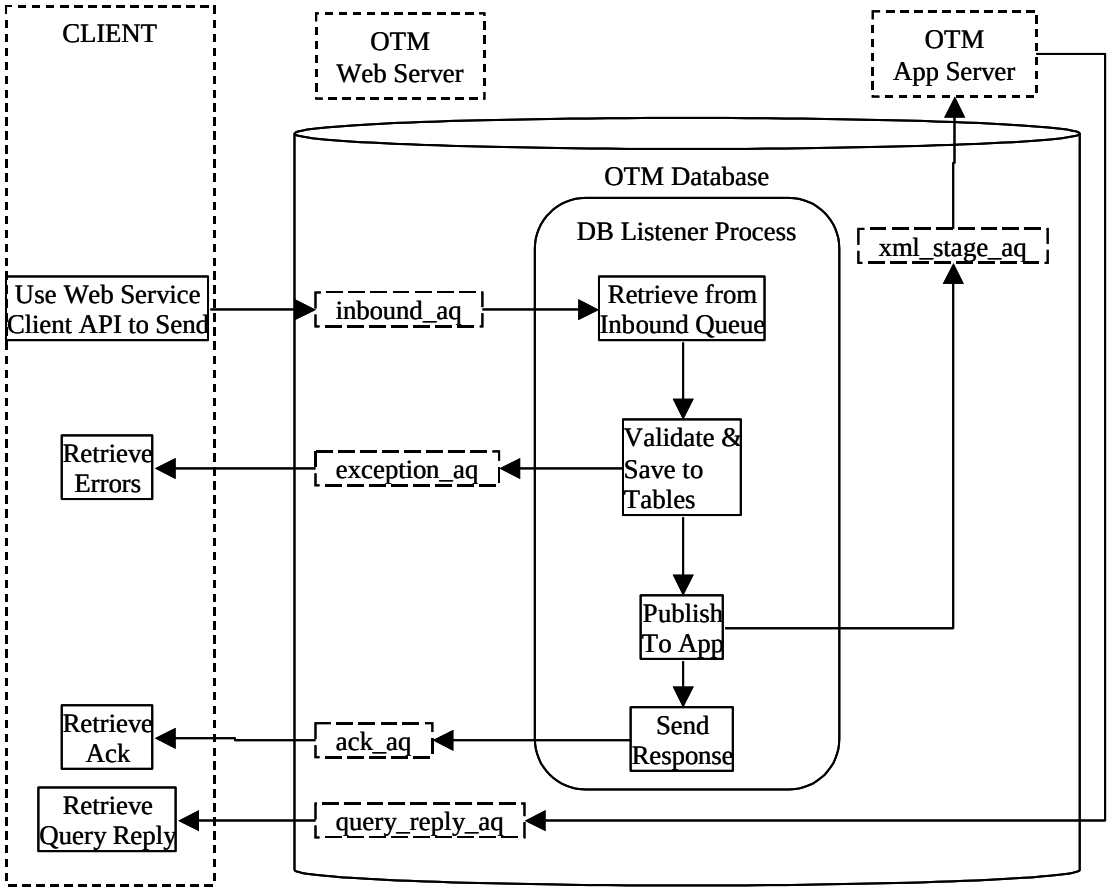


Figure 9.1

In the diagram above, you first send the XML to the Inbound Queue. The database listener reads from the "Inbound Queue" and stages the data to the i_transmission/i_transaction tables. If staging is successful, the database listener puts the TransmissionAck message in the "Ack Queue" and stages a message to the application server in the "XML Topic Queue" so that the application server can proceed with processing the message. If the Transmission XML is for a RemoteQuery, the query response is placed in the query_reply_aq queue. If an error occurred in the staging, the database listener puts an exception message in the "Exception Queue". The configuration of the application server listener is discussed later. All the queues may or may not be on the same queue table. However, the "Inbound Queue" and "XML Topic Queue" must not be multi-consumer queues. Furthermore, you are allowed to add multiple sets of inbound and outbound queues in the same queue table.

To create all queues required for inbound processing on the same queue table, use the following procedure:

```
procedure setup_inbound_queue_system(p_queue_table_name varchar2,
   p_inbound_queue_name varchar2,
   p_xmltopic_queue_name varchar2,
   p_ack_queue varchar2,
   p_exception_queue varchar2)
```

Example:

```
Sqlplus> execute pkg_queue_management.setup_inbound_queue_system
('queue_test_table',
 'another_inbound_aq',
```



```
'raise_xml_topic',
'acknowledgement',
'notify_exception');
```

To create the queues on different queue table(s), use the following procedure:

```
procedure start_queue(p_queue_name varchar2, p_queue_table_name varchar2)
```

Example - to create all the queues in example A above individually, use the following commands:

```
Sqlplus> execute pkg_queue_management.start_queue ('another_inbound_aq',
'queue_test_table');
Sqlplus> execute pkg_queue_management.start_queue( 'raise_xml_topic',
'queue_test_table');
Sqlplus> execute pkg_queue_management.start_queue ( 'acknowledgement',
'queue_test_table');
Sqlplus> execute pkg_queue_management.start_queue ( 'notify_exception',
'queue_test_table');
```

Step 3 – Setup Database Listeners

For each inbound set of queues created above, a database listener should be created in order for the XML to be processed. The following procedure is used to setup the listener:

```
procedure install_queue_listener(p_inbound_queue_name varchar2 ,
   p_xmltopic_queue_name varchar2 ,
   p_ack_queue varchar2 ,
   p_exception_queue varchar2)
```

Example:

```
Sqlplus> execute pkg_queue_management.install_queue_listener  (
'another_inbound_aq',  'raise_xml_topic', 'acknowledgement',
'notify_exception');
```

In this case, you first send the XML to the 'another_inbound_aq' queue defined in the first parameter. The database listener reads from this queue and stages the data. If staging is successful, the database listener puts the TransmissionAck message in the 'acknowledgement' queue defined in the third parameter, and stages a message to the application server in the 'raise_xml_topic' queue defined in the second parameter. If an error occurred in the staging, the database listener puts an exception message in the'notify_exception' queue defined in the fourth parameter.

To stop the database listener, execute the following procedure on the "Inbound Queue":

```
Sqlplus> execute
pkg_queue_management.stop_queue_listener('another_inbound_aq');
```

Step 4 – Setup Application Server Listeners

After the database listener successfully stages the XML, it submits a message to the "XML Topic Queue" for the app server to process. The application server requires a listener/thread to be enabled to process the messages in the "XML Topic Queue". The app server listener is set up through properties. The format of property entry is (*note the difference in glog.integration.oaq vs. glog.oaq.integration*):

```
glog.oaq.integration.{the_topic_queue_name}=1
```


For example, the property entry corresponding to the database listener created in (3) should be:

```
glog.oaq.integration.raise_xml_topic=1
```

The value for the property must be a non-zero integer. The integer value determines the total number of threads for the listener. Since the application server listener is very lightweight, one thread should be enough to process the messages. If user desires to set up the value greater than one, a performance test should be done to determine the effects. To turn off the listener, set the value to "0" or remove the property entry. The property only takes effect during the startup.

Auto Startup of Database Listener via Application Server

The app server has the ability to start and stop the database listener when it is being started or shut down. This is enabled through the use of the following property:

```
glog.integration.oaq.controlDbListener=true
```

When the property is true, the application server will also start the database listener when the app is starting, and will also shut down the database listener when the app server is shutting down.

Backward Compatible Application Server Properties

Prior to Oracle Transportation Management Release 5.0, the application server listener was started by setting the property glog.integration.oaq=true. Note that this property is deprecated. The suggested property is "glog.oaq.integration.xml_stage_aq=1". For backward compatibility, the property "glog.integration.oaq=true" is still supported and correlates to enabling the suggested property "glog.oaq.integration.xml_stage_aq=1".

Step 5 – Create Outbound Queues

You specify the queue to use for sending outbound XML from Oracle Transportation Management in the External System Manager in the UI. There are two approaches for creating the outbound queue, which is then used in the External System Manager. The first approach is to create the queue using the stored procedure; this enables you to specify the queue table to be used for the queue. After the queue is created, the external system can then reference the queue. The second approach is to specify the queue in the external system manager without first creating the queue. If the queue does not exist, the Oracle Transportation Management application would create the queue with the queue table defined in the property entry glog.integration.oaq.outbound.queuetable. By default, the queue table is intg_queue.

Example to create queue from procedure:

```
Sqlplus>  execute pkg_queue_management.start_queue ('outbound_example_queue',
'outbound_queue_table');
```

If the queue table is a multi-consumer queue table, the corresponding queue on the table is multi-consumer. At least one subscriber must be created for the queue; otherwise, Oracle Transportation Management will throw an exception during the enqueue process.

The following procedure will add a subscriber to the multi-consumer queue:

```
Sqlplus>   Pkg_queue_util.add_subscriber('mutlti_consumer_queue',
subscriber_name);
```


Step 6 – Other Queue Management Utilities

To drop a queue:

```
execute pkg_queue_management.drop_queue ('your queue_name');
```

To delete all queue entries for a given queue:

```
execute pkg_queue_management.delete_queue_entries('your queue_name');
```

To remove every entry for all the queues in a given non multi-consumer queue table:

```
execute pkg_queue_management.empty_queue_table('queue table name');
```

To drop all queues in a given table:

```
execute pkg_queue_management.drop_all_queues('queue table name');
```

To drop a queue table as well as the corresponding queues:

```
execute pkg_queue_management.drop_queue_table('queue table name');
```

To stop all database listeners:

```
execute pkg_queue_management.stop_all_queue_listeners;
```

To stop a specific database listener:

```
execute pkg_queue_management.stop_queue_listener('inbound queue');
** The "inbound queue" is the first parameter in install_queue_listener
```

To remove database listeners:

```
execute pkg_queue_management.remove_all_queue_listeners;
```

To remove a specific database listener:

```
execute pkg_queue_management.remove_queue_listener('inbound queue');
```

Optional Oracle Settings

The following Oracle parameters can be specified in init.ora or spfile. Refer to Oracle database documentation for additional details on these parameters.

- aq_tm_process = 1 (to perform time monitoring on queue messages)
- job_queue_processes = 6 (to set the number of job queue processes started in an instance)

The following sections highlight some control features that can be implemented with the OAQ runtime functionality.

Correlation of TransmissionAck to Transmission

Because of the asynchronous nature of message queues, the TransmissionAck that is placed in the "Ack Queue" or the "Exception Queue" may need to be correlated to determine the original Transmission that is referred to. There are several options available to provide this correlation:

- Use Sender Transmission Number: You must send a unique value in the TransmissionHeader.SenderTransmissionNo element in the Transmission XML before sending it to the queue. The value is echoed back in the TransmissionAck and the TransmissionReport.
- Use refnum field in INTG_QUEUE_MESSAGE: You must set the refnum field in the queue message before queuing. The value is echoed back as the SenderTransmissionNo in the TransmissionAck and the TransmissionReport.
- Limited Use of JMSCorrelationID Header Option: A column exists in the queue table for a correlation id (table column name is CORRID). This field can be populated on the inbound queue to Oracle Transportation Management, but is not populated on the queue table for the acknowledgement or the report. When this field is populated on the inbound queue, it is mapped to the Sender Transmission Number. The value is echoed back in the TransmissionAck and the TransmissionReport.

Suppression of TransmissionAck

When using OAQ, clients can rely on the confirmation of receipt of the Transmission via the successful enqueing of the Transmission. With this approach, the TransmissionAck response is redundant. It is possible to suppress the TransmissionAck as a response by setting the SuppressTransmissionAck element to Y in the TransmissionHeader element. Errors that may occur with receiving the Transmission will still be reported in the error queue.

TransmissionReport Sent Via QUEUE

After the application server completes processing of a Transmission, a TransmissionReport can optionally be sent to a recipient indicated the status of processing. It is also possible to place this TransmissionReport into a queue by setting the following in the TransmissionHeader:

```
AckSpecComMethodGID = 'QUEUE'
```

The AckSpec.ContactGID element should be specified. Oracle Transportation Management will look for Contact > External System > IntQueueName. The process also supports specifying a ContactGID for which the TransmissionReport should be sent. The ComMethodGID specifies the method of sending.

> **Note**: The correlation of the TransmissionReport to the original inbound Transmission is accomplished by specifying the SenderTransmissionNo in the inbound Transmission.

10. Integration Data Queues

Overview

By default the Oracle Transportation Management application server workload is managed by a set of in-memory event queues; with a configurable number of threads available to process each queue. The Oracle Transportation Management Data Queue infrastructure was introduced in version 5.5 to permit a more configurable fine-grained control of various aspects of this internal workload. The Data Queues are database resident queues where a **configurable** number of "poller" threads retrieve a **configurable** number of events at a **configurable** interval and pass them to an "executor" to be processed. See Figure 10.1 below.

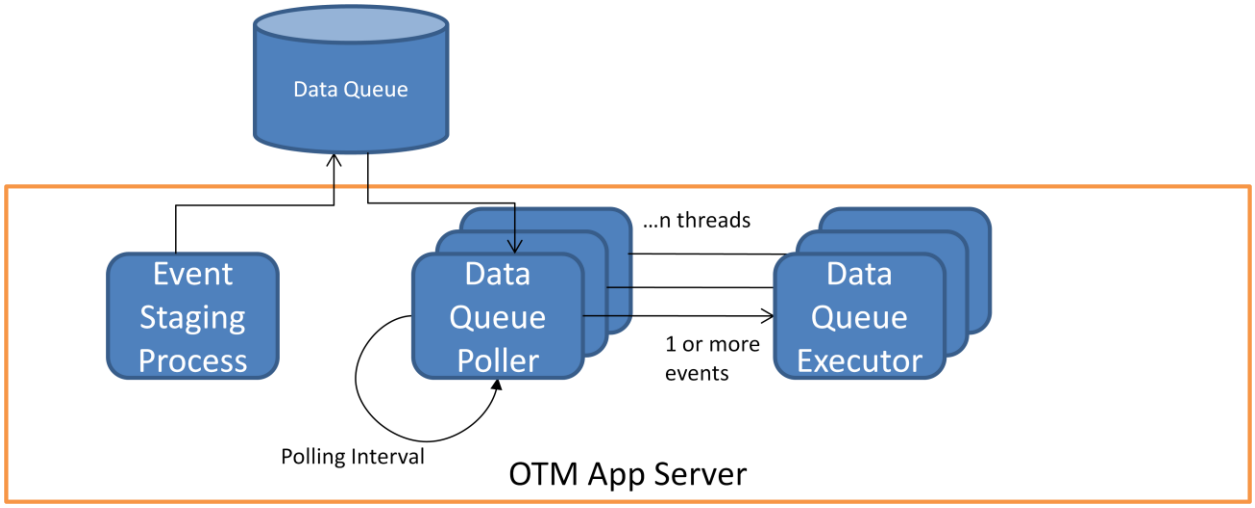


Figure 10.1

> **Note:** Although the entity names used are similar, Oracle Transportation Management Data Queues do not use Oracle Advanced Queue objects for processing data queue events.

The internal processes for Inbound and Outbound Integration can now be configured to use this infrastructure.

> **Note:** See *Data Queue Manager* in the OTM online Help for coverage on the User Interface used to configure each data queue.

Figure 10.2 shows the application server processing of an inbound Shipment Status XML without the use of data queues. The XML message will arrive via one of the supported protocols: HTTP, SOAP Web Service or OAQ.

> **Note:** the Direct Insert XML protocol is covered in the Direct XML Insert section.

1. The XML is stored in the Transmission table with a new unique Transmission number.
2. A New XML topic containing the Transmission number is placed on the in-memory event queue.
3. A listener thread which has subscribed to New XML topics removes the event and passes it to the New XML Workflow to process.
4. Workflow processes the content of the XML, which in this case results in a new status being assigned to a shipment.

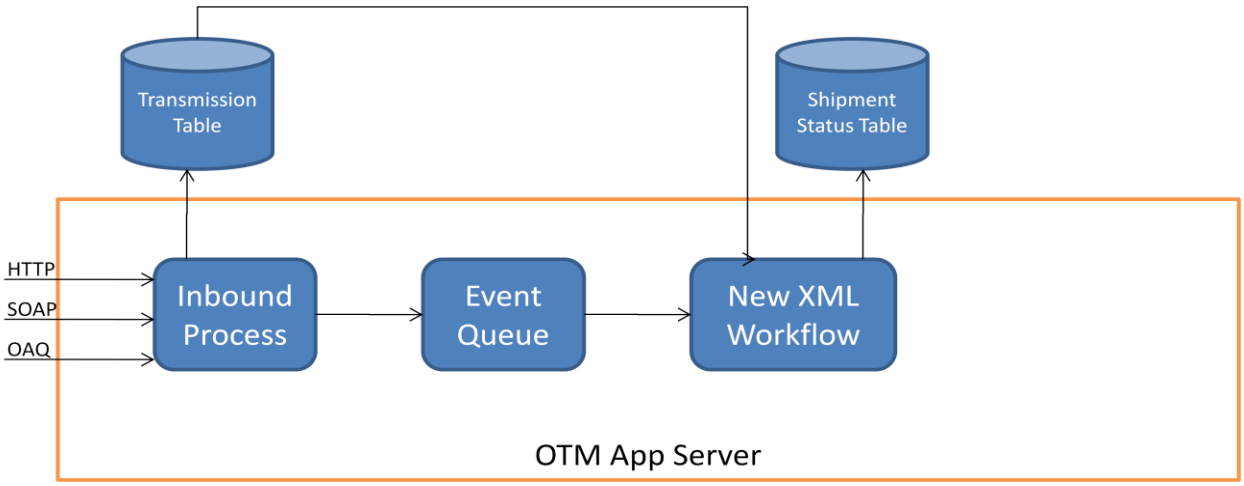


Figure 10.2

Imagine an Oracle Transportation Management implementation scenario that has an inbound interface with a carrier status application where shipment statuses are sent in one batch of 10000 to Oracle Transportation Management. Without Data Queues, the application server in-memory event queues would contain all 10000 Shipment Status transactions in one backlog, therefore competing with all other internal application server processes for resources and process time.

Figure 10.3 shows the application server processing where data queues were configured for inbound XML messages. The Inbound Data Queue process can instead be configured to stage the initial 10000 transactions and process in batches of 1000 every 5 minutes.

1. The XML is stored in the Transmission table and the associated Transmission number is stored in a Data Queue Event.
2. At some time within the configured interval, a Poller thread will retrieve up to 1000 data queue events and pass individually to the Executor to be processed.
3. The Executor publishes a New XML topic which, from then on, will be processed as with normal in-memory event queues.

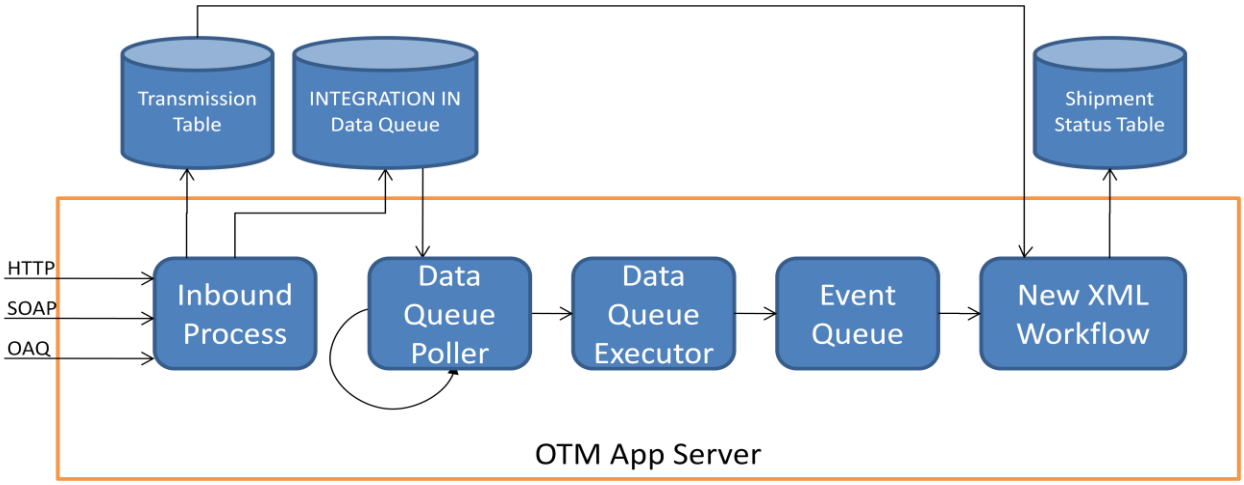


Figure 10.3

The application server processing for Outbound XML messages also uses in-memory event queues to manage workload; one for the building of the XML and another for the physical transport of the message. The use of Data Queues for outbound message works essentially identical to the inbound

scenario described above i.e. instead of publishing new build or transport topics on in-memory event queues, data queue events are stored in outbound data queues. The same arrangement of Poller and Executor then publishes the topic to the next step in the process.

Activating Integration Data Queues

There are several Integration Data Queues available in the PUBLIC domain:-

- **INTEGRATION IN**: Used to manage all Inbound GLogXML Transmissions sent in via UI (Upload Transmission), HTTP (WMServlet), SERVICE (IntXmlService), and OAQ (INBOUND_AQ).
- **INTEGRATION IN DIRECT XML:** (covered in **Direct XML Insert** section.)
- **INTEGRATION OUT XML BUILD**: Used to manage outbound process of generating Transmission content from database objects and staging in Transmission tables ready for transport.
- **INTEGRATION OUT TRANSPORT – HTTP**: Used to manage transmissions ready to be transported via HTTP to external system.
- **INTEGRATION OUT TRANSPORT – SERVICE**: Used to manage transmissions ready to be transported via web service call to external system.
- **INTEGRATION OUT TRANSPORT – FTP**: Used to manage transmissions ready to be transported via FTP to external system.
- **INTEGRATION OUT TRANSPORT – QUEUE**: Used to manage transmissions ready to be transported via Oracle Advance Queue to external system.

The inbound data queue events reside in the Q_INTEGRATION_IN Data Queue table. The outbound data queue events reside in the Q_INTEGRATION_OUT Data Queue table.

By default the PUBLIC queues are inactive. To activate the queues, login as DBA.ADMIN and edit the Data Queue records via the Data Queue Manager (under **Business Process Automation > Integration** menu).

> **Note**: It is not recommended that you activate or customize any of the public data queues. If you need to activate or customize a data queue, copy the queue and edit the copy. PUBLIC queues should be used as a template to create your own queues. PUBLIC data is not static and may change between releases. Custom changes should be made in domain-specific queues, not public ones. If you customize one of the integration queues, you must configure the appropriate glog.integration.dataqueue property so that it references the name of the customized data queue.

Alternatively, the PUBLIC records can be copied to DOMAIN specific records and activated there. In order for the application server to recognize the new queue names, the following properties must also be set:

```
glog.integration.dataqueue.inbound
glog.integration.dataqueue.xmlBuild
glog.integration.dataqueue.transport.http
glog.integration.dataqueue.transport.service
glog.integration.dataqueue.transport.ftp
glog.integration.dataqueue.transport.queue
```

For example,

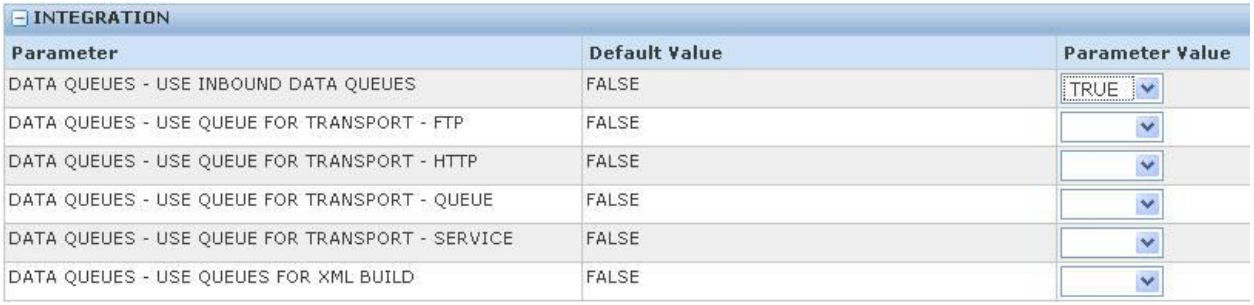
```
glog.integration.dataqueue.inbound=MYDOMAIN.MYINBOUNDQ
```


Note: An application server can only process one data queue of each type. For example it is not possible to configure two inbound data queues: MYDOMAIN.INBOUNDQ1 & MYDOMAIN.INBOUNDQ2 or MYDOMAIN2.INBOUNDQ1. The only exception to this is the data queue configuration for Direct Insert XML. See the Direct XML Insert section for details.

Once one or more Integration Data Queues are active, the processing can be controlled on a domain-by-domain basis by setting one of the following PARAMETER SET parameters to 'TRUE' on the DOMAIN PARAMETER SET.

- DATA QUEUES - USE INBOUND DATA QUEUE
- DATA QUEUES – USE QUEUE FOR XML BUILD
- DATA QUEUES – USE QUEUE FOR TRANSPORT – HTTP
- DATA QUEUES – USE QUEUE FOR TRANSPORT – SERVICE
- DATA QUEUES – USE QUEUE FOR TRANSPORT – FTP
- DATA QUEUES – USE QUEUE FOR TRANSPORT – QUEUE

For example,



If either the queue is not active or the domain parameter is FALSE, the default in-memory queues will be used.

Note: It is possible to override this default to process via in-memory queues. Set property `glog.integration.dataqueue.inbound.useMemoryQueue=false`. This will cause the inbound Transmissions to remain in 'STAGED' status.

Understanding the Integration Data Queue Definitions

The standard installation of OTM version 6.2 has PUBLIC Data Queue Definitions configured for each of the Integration Data Queues described in the preceding section.

This section will describe these definitions so that any customization can be done without any unforeseen impact on the integrity or performance of the OTM servers.

The Data Queue Definition is a unique (based on a Data Queue Definition GID) configuration of the following properties.

Parameter	Description
Active	Only queues which are marked as 'Active' will have Poller threads started on the application server.

Parameter	Description
Thread Count	The number of Poller threads that will be started on each application server. All Poller threads will synchronize their access to the events on a data queue. One thread at a time will retrieve a batch of events, mark them all as 'ACTIVE' and commit the transaction before releasing the queue for another thread to access. **This would normally be tuned to match the expected volume of events.**
Batch Size	The maximum number of events that will be retrieved by each Poller thread. **This would normally be tuned to match the expected volume of events.**
Polling Frequency	The duration each Poller thread will sleep between attempts to retrieve a batch of events. **Note:** if the time taken to process a batch exceeds the polling frequency the Poller thread will essentially start processing the next batch more or less immediately after completing the current batch. **This would normally be tuned to match the expected volume of events.**
Data Queue Table	Corresponds to the physical table which will contain the events **This should not require customization.**
Data Queue Poller	The Poller configuration specifies the behavior and attributes of the SQL statement used to retrieve the batch of events. In addition to the Class that will execute the query, the following can be customized:- • Filter – these are additional predicates added to the basic statement (e.g. status = 'QUEUED' etc.) • Order – this is specified as the ORDER BY criteria. This is how priority ordering is achieved. Poller Definitions are configured via Business Process Automation Power Data. **Custom Poller definitions may be required if additional criteria is used to further partition events placed on the same queue e.g. by joining other tables.**

Parameter	Description
Data Queue Executor	The Executor configuration species the Class that will process the actual event and whether events in the ACTIVE or FAILED state should be reprocessed on app server startup. Executor Definitions are configured via Business Process Automation Power Data. **Custom Executor definitions may be required if reprocessing of events needs to be handled differently.**
Finder Set	This specifies the Class which formats the queue specific information in the Data Queue Manager Events UI. **This should not require customization.**
Related Queue	This is used to manage the parent/child relationship between events. **This is not used by Integration Data Queues.**

Customizing Inbound Data Queue Definitions

There are two inbound data queue definitions created by default in the PUBLIC domain: INTEGRATION IN and INTEGRATION IN DIRECT XML. Events for both are held in the Q_INTEGRATION _IN Data Queue Table. By default both queues are inactive.

If these base configurations are deemed to be sufficient for the expected volumes, they can be activated by logging in as DBA.ADMIN and setting each queue definition to 'Active'. Additionally, as mentioned previously, the INTEGRATION IN Parameter set value also needs to be set to 'TRUE'.

Customized Poller Definition

The Poller Definitions for both queues are configured to use the PREEMPTIVE POLLER plug-in, which uses a 'Top N query' format to retrieve an ordered batch of events. The format of the SQL statement that would be used to retrieve a batch of events would essentially be as follows:-

```
SELECT rownum, {column list}
   FROM (SELECT {column list}
            FROM Q_INTEGRATION_IN
            WHERE q_state='QUEUED' and {filter}
            ORDER BY {order})
   WHERE rownum <= {batch size}
```

The arguments in braces are replaced at runtime as follows:-

Parameter	Description
column list	The complete column list is controlled via the Event Class which for Q_INTEGRATION_IN events would be:- `q_event_seq, rownum, q_start, q_process_start, domain_name, q_preemption_priority, topic_class_name, topic_parameters, topic_description, i_transmission_no, q_queue_name.`
filter	The PUBLIC Poller definitions for INTEGRATION IN and INTEGRATION IN DIRECT XML has a filter clause as follows:- q_start < sys_extract_utc(current_timestamp) and q_queue_name = 'INTEGRATION IN' or q_start < sys_extract_utc(current_timestamp) and q_queue_name = 'INTEGRATION IN DIRECT XML' Therefore, if a custom Data Queue Definition is to be used there will need to be a corresponding custom Poller Definition in order to override the `q_queue_name` parameter.
order	For Preemptive Data Queues the base ORDER BY clause will be as follows:- q_preemption_priority, q_event_seq That is, ordered by priority first and then event sequence number.
batch size	This is the batch size taken from the Data Queue Definition.

Customized Executor Definition

In addition to the ability to configure redo of ACTIVE or FAILED events, it is possible to configure how the topics created by processing Direct Insert XML events are processed. See the Direct XML Insert section for details.

Customizing Outbound Data Queue Definitions

The Outbound data queue definitions are subjected to the same customization capabilities and constraints as the inbound data queues.

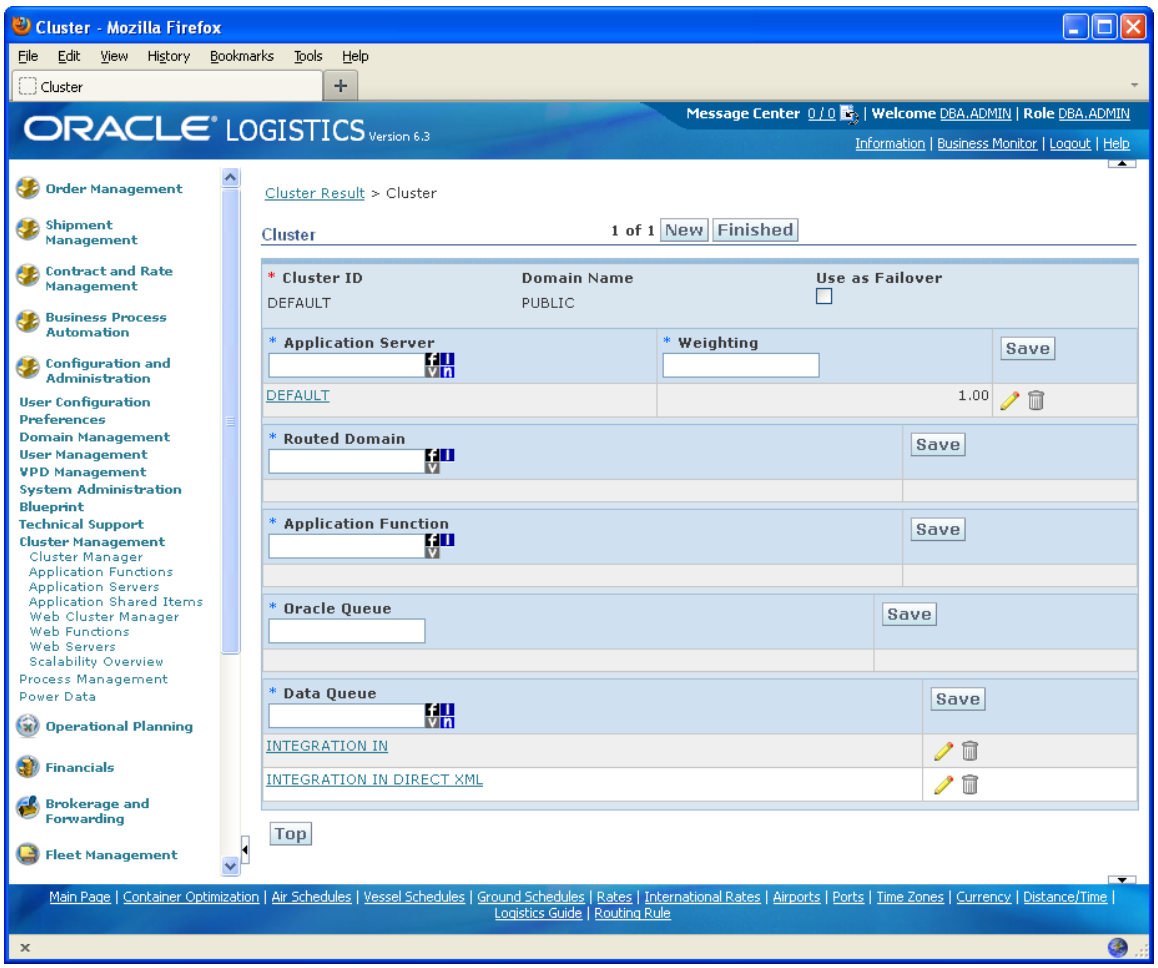
Scalability & Clustering Considerations

In a Scalability environment there are additional considerations required to ensure that any required Data Queues are active and processing events as and when required.

Poller threads will only be started on Application Servers which belong to the Cluster associated to a particular data queue. Data Queues are associated to a Cluster via the Cluster Manager under Configuration and Administration, Cluster Management.

For example, assume CLUSTER-01 is defined and has app servers APP-01 and APP-02 assigned to it via the Cluster Manager. App server APP-03 is assigned to CLUSTER-02.

If data queue definition INTEGRATION IN is also assigned via the Cluster Manager to CLUSTER-01, then Poller threads will started when the APP-01 and APP-02 app server machines are started. Poller threads will not be started on APP-03 and therefore it will not process any Data Queue events for INTEGRATION IN.



Monitoring Integration Data Queues

In addition to its primary task of administering the data queues themselves, the Data Queue Manager is also used to monitor the activity of events on each queue via the Events action button. This provides access to a Finder/Results screen from where all events for a queue can be listed and individual event details can be viewed. The majority of information is generic to all Data Queues e.g. Process Time, and Log Process ID. The following sections describe the Integration Data Queue specific data.

Inbound Events

All inbound events have an associated transmission number. In normal processing, when a transmission is sent to Oracle Transportation Management, the XML is persisted in the transmission tables with an initial status of STAGED and a New XML event is published almost immediately. Once the New XML event execution has started the status is changed to FRESH. When the Inbound Data Queue is active, the XML is persisted in the transmission tables with a status of STAGED and the New XML event is persisted in the data queue table. Consequently, transmissions can remain in this STAGED status for much longer.

Inbound events can also be preempted. In other words, events already present in the data queue can be selected for processing ahead of other events that may have been inserted before it. By selecting an event for preemption, the Q_PREEMPTION_PRIORITY is set to the lowest value in the data queue table. This will ensure the next POLLER will retrieve the selected event.

Outbound Events

Outbound events may go through two event queues: XML Build and Transport. The XML Build process happens before a transmission record has been created and so these events do not have an available transmission number. They do, however, have a Notification Context that shows the object type of the communication, such as Location, and one or more object GIDs.

The display of the context is limited to approximately 4000 characters and so if the context is greater than this it will not be displayed.

As with inbound events, outbound events can be preempted.

However, unlike inbound events, outbound event priority can be set when the data queue event is staged in the data queue table. The priority value is managed via the External System where it can be set alongside an OUT XML PROFILE for a particular GLogXML Element e.g. PlannedShipment.

11. Direct XML Insert

Overview

Direct XML Insert is a new way of sending GLogXML Transmissions and Mobile Communication Message XML to Oracle Transportation Management. The documents are inserted directly into the DB via a new PL/SQL procedure – `insert_transmission` or `insert_message` – in the `pkg_integration_util` package.

> **Note:** The previous method of inserting via the NEW_XML_TRANSMISSION view has been deprecated and will be removed in a future release.

Database User: DIR_XML_USER

This user has all required privileges and synonyms to successfully insert transmission into the database. Use DIR_XML_USER to execute these procedures.

The procedure signatures are described below:

```
insert_transmission (   p_username IN VARCHAR2,
                        p_password IN VARCHAR2,
                        p_transmission IN CLOB,
                        p_transmission_no OUT NUMBER,
                        p_data_queue IN VARCHAR2 DEFAULT NULL,
                        p_cluster_gid IN VARCHAR2 DEFAULT NULL,
                        p_priority IN NUMBER DEFAULT 0)
```

and:

```
insert_message (        p_username IN VARCHAR2,
                        p_password IN VARCHAR2,
                        p_message IN CLOB,
                        p_message_gid OUT VARCHAR2,
                        p_data_queue IN VARCHAR2 DEFAULT NULL,
                        p_cluster_gid IN VARCHAR2 DEFAULT NULL,
                        p_priority IN NUMBER DEFAULT 0)
```

Parameter	Description
p_username	**IN.** OTM user account under who's authority XML will be processed.
p_password	**IN.** Password for OTM user.
p_transmission {insert_transmission}	**IN.** Transmission XML message.
p_message {insert_message}	**IN.** Message XML message.
p_transmission_no	**OUT.** Unique Transmission number generated for Transmission XML message.

Parameter	Description
p_message_gid	**OUT.** Unique Message GID generated for Message XML message.
p_data_queue	**IN. Default NULL.** Data Queue Definition GID that event will be handled by.
p_cluster_gid	**IN. Default NULL.** Cluster that will process the events. If this parameter is specified on insert then the property `glog.dataqueue.eventCluster` must be set equal to the Cluster GID on all Application Servers in the cluster which should process the events.
p_priority	**IN. Default 0.** Priority of event. Used by Poller to order sequence of events for Preemptive Poller.

Internal Processing

The Transmission or Message XMLs inserted using these procedures are processed using an internal Integration Data Queue – INTEGRATION IN DIRECT XML and can be monitored in exactly the same way as Integration Data Queue events covered in **Integration Data Queues** section.

Figure 11.1 shows the processing for directly inserted XML messages.

1. XML message passed to `insert_transmission` or `insert_message` procedure.
2. OTM procedure stores XML in Transmission table, data queue event in Direct Insert Data Queue and returns Transmission number to caller.
3. The Poller retrieves a number of events and calls the Executor for each event.
4. The Executor retrieves the message XML and calls the New XML Insert Workflow to perform the following Transmission pre-processing:
    - Populate Transaction table data
    - Populate Process Group data if present.
    - Populate Transmission reference numbers if present.
    - Publish New XML topic or queue event to INTEGRATION IN data queue if configured.

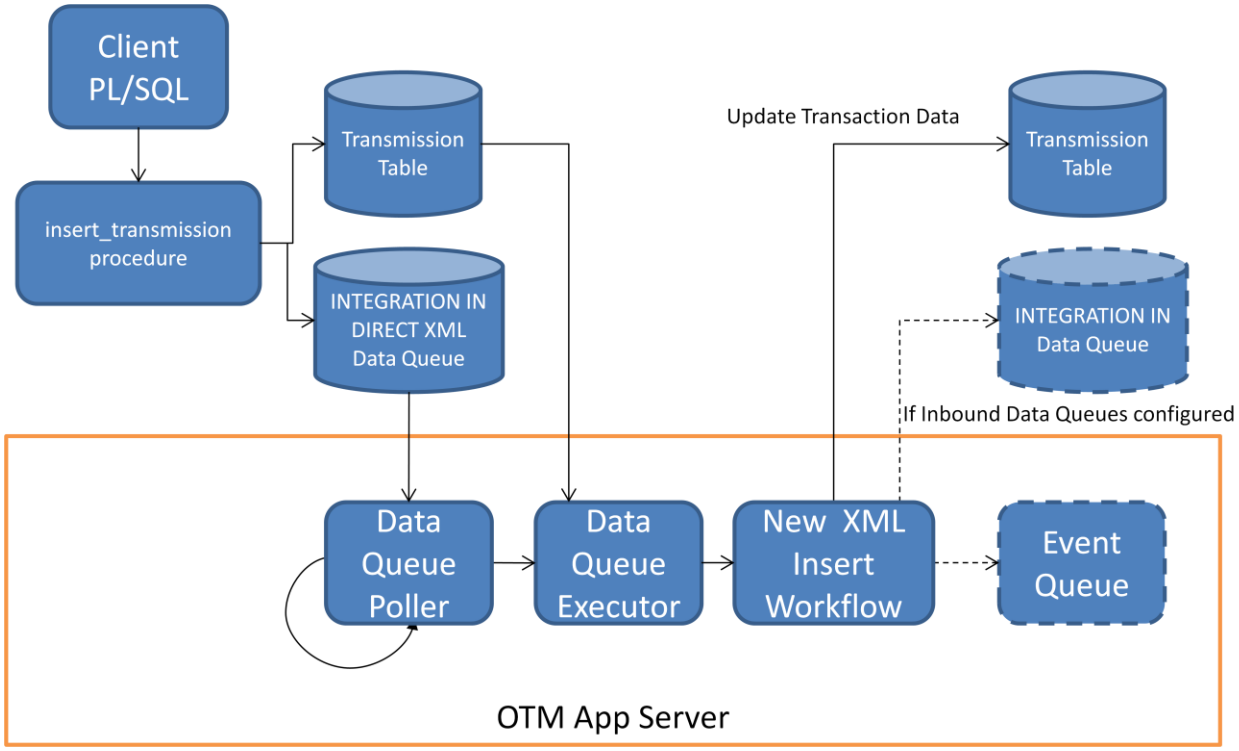


Figure 11.1

Data Queue Definition Customization

The INTEGRATION IN DIRECT XML Data Queue can be customized exactly as described in the Customizing Inbound Data Queue Definitions section, with the following additions.

1. Direct XML supports multiple Data Queue Definitions
    a. Whereas the other Inbound and Outbound Data Queue Definitions to be used are determined by logic in the application server, the Direct Insert XML Data Queue GID can be selected by the client and specified as a parameter in the procedure call to insert the XML message.
    b. Multiple Data Queue Definitions allow the ability to partition a large volume of XML messages into separately controlled groups as each Definition can specify distinct values for number of threads, batch size and polling interval.

    Note: Multiple Data Queue Definitions would also require multiple Poller Definitions because the queue name (i.e. the GID) forms part of the Poller Filter and so would need to be customized.

2. Multiple Executor options used to process events
    a. By default the executor used to process the initial event on the INTEGRATION IN DIRECT XML data queue is the INTEGRATION IN DIRECT XML executor as described in Figure 11.1. If the INTEGRATION IN data queue is also active, this will mean that a Transmission sent in using Direct Insert XML will pass through two data queues. An alternative executor is available - INTEGRATION IN COMPOSITE – which will instead by pass the INTEGRATION IN data queue (i.e. as if it were not active) and proceed directly to the New XML Workflow. This is useful in scenarios where multiple transport methods are used e.g. Web Service and Direct Insert XML but it is only desired to use one data queue for each workload.

b. If using a prioritized poller, the use of the INTEGRATION IN COMPOSITE executor also means that the assigned priority causes the New XML workflow to be triggered in priority order. However, the priority is not used beyond this point i.e. the internal logic of the New XML Workflow no longer uses the priority after it is first triggered.

3. Method used to call New XML Insert Workflow is configurable

   a. By default, the New XML Insert Workflow is called by the INTEGRATION IN DIRECT XML Executor using the 'execute' paradigm, which means the workflow is executed using the Poller thread which retrieved the event. This method can be changed to either 'publish' or 'publishWait' by setting the property `glog.integration.dataqueue.inboundDXIType` which if not set defaults to 'execute'.

      i. **Execute**: executed using the Poller thread which retrieved the event.

      ii. **Publish**: topic is published to in-memory event queue and process at some time later by another thread. Poller thread is free to process another event without waiting.

      iii. **publishWait**: topic is published to in-memory event queue and processed at some time later by another thread. The Poller thread waits for the topic processing to complete before attempting to process another event.

12. Oracle Transportation Management Web Services

Web Service Security (WS-Security)

The Web Service Security Specification is an OASIS standard for defining security related information as part of a SOAP message. See http://www.oasis-open.org/.

In previous releases, Oracle Transportation Management only supported the WS-Security Username Token Profile. Furthermore, the declaration of this support had to be made in the Oracle Transportation Management database via the Web Service Manager User Interface.

As of version 6.3, support is now available for Web Service Policy (WS-Policy) assertions for the following WS-Security related features (referred to as WS-SecurityPolicy):-

- Username Token Profile
- HTTPS Transport
- Message Encryption

  Note: Please refer to the Oracle Transportation Management Security Guide for configuration details for using Web Service Security for inbound and outbound messages.

Agent Action Web Services

The following web services are available from version 6.0 onwards:

- Execute individual agent action against one or more objects of a supported type.
- Execute multiple agent actions against one or more objects of a supported type.

Currently the supported object types are SHIPMENT, SELL SIDE SHIPMENT, ORDER RELEASE, and ORDER MOVEMENT.

Therefore, the following services are available:

- ShipmentService
- SellSideShipmentService
- OrderReleaseService
- OrderMovementService
- AgentService

The type specific services handle individual agent actions for that type and have one operation: `processAction`. The AgentService handles execution of multiple actions essentially identical to an Oracle Transportation Management Agent but with some constraints (covered later) and also has one operation: `processAgent`.

All services are implemented with the synchronous REQUEST/RESPONSE messaging model. However, with respect to Oracle Transportation Management agent action processing, the response indicates that the action has been scheduled successfully. This is due to the fact that the Oracle Transportation Management application could have a significant amount of workflow triggered by such an action. Therefore waiting for completion may require an excessive transaction timeout value.

The input and output messages for each service are specified in the service XSDs. Namely:

- AgentService.xsd
- ShipmentService.xsd

- SellSideShipmentService.xsd
- OrderReleaseService.xsd
- OrderMovementService.xsd

Agent specific message content is specified in the following XSDs:-

- **Agent.xsd**: contains agent header details
- **ShipmentAction.xsd**: contains all actions related to buy shipments
- **SellSideShipmentAction.xsd**: contains all actions related to sell shipments
- **OrderReleaseAction.xsd**: contains all actions related to order releases
- **OrderMovementAction.xsd**: contains all actions related to order movements

The WSDL for all services is available via the Retrieve WSDL servlet discussed in the **Servlet for Accessing Web** Services Description Language (WSDL) Files section.

> **Note:** Due to a change in v 6.2 in how these services are deployed in the application server at runtime, the WSDL for each service will not directly reference the XSD files mentioned above. However, the formats defined in these XSD files will be functionally equivalent to the definitions referenced in the WSDL.

AgentService constraints

Although in theory a complete agent could be defined using the schemas, some actions will not be supported in the initial version. These actions are known as the BLOCK actions: IF, ELSE, FOR EACH, etc.

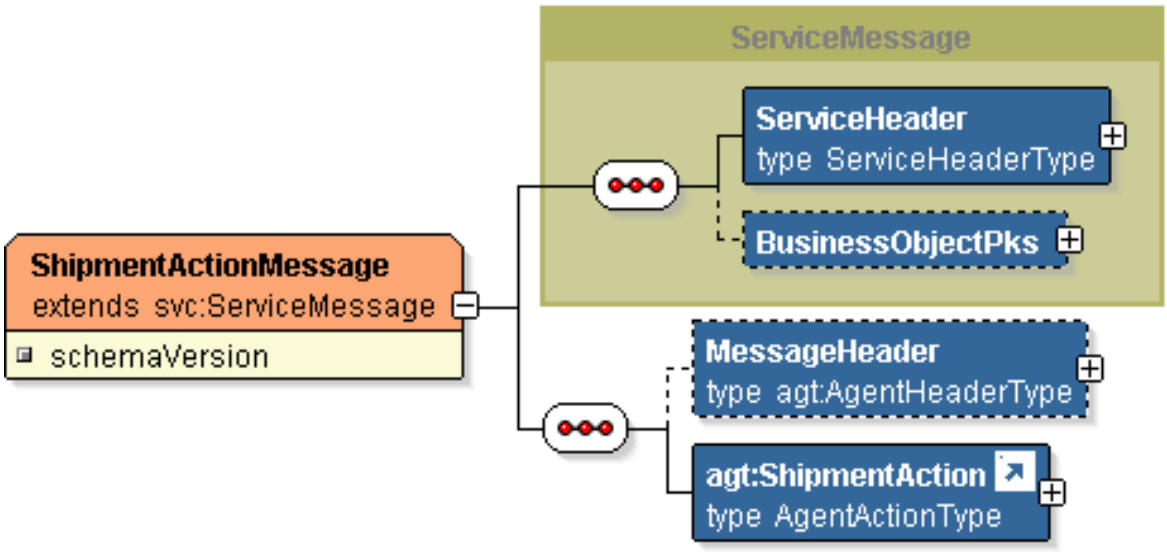
Version Control

The service and message definition schemas are under version control starting from version 1.0, i.e. major version number is 1, and minor version number is 0. The current target namespace for each schema will contain the major version number, for example the target namespace for version 1.0 of Agent.xsd is:

http://xmlns.oracle.com/apps/otm/agent/v1

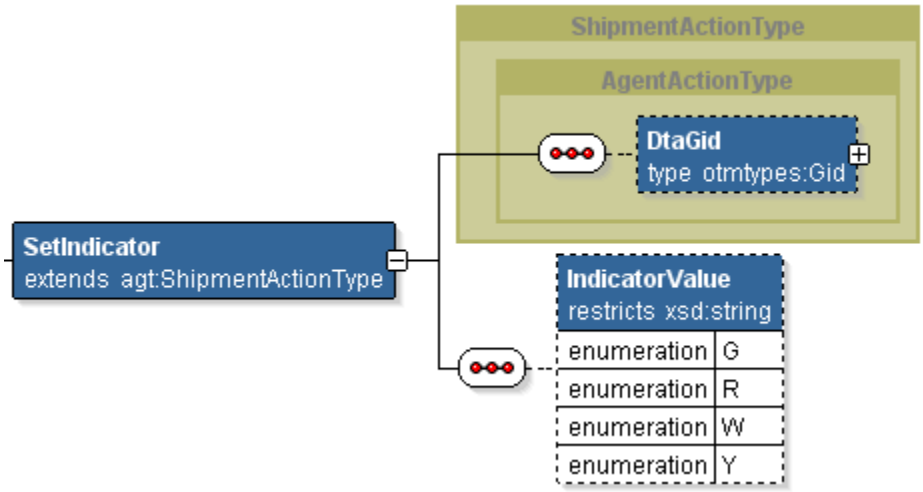
As of version 6.2, the current version of the Agent Action web services has increased to "v2". However, to support backward compatibility, all "v1" messages will still be valid.

Example Service Message



All service messages extend the `ServiceMessage` complexType (defined in the Service.xsd schema).

The Agent Action messages then include Agent Header information (`MessageHeader`) followed by an Agent Action, in this example `ShipmentAction`. All valid Shipment Actions are defined in the ShipmentAction.xsd schema. The following example shows the Set Indicator Action:



The `SetIndicator` element is allowed to be substituted for the `ShipmentAction` element because it is defined as an XSD **substitutionGroup** and extends the same `AgentActionType` complexType.

Changes between Version 1.0 and Version 2.0

Optional Username and Password elements

The initial version of the Service XSD schema had the username and password element as "required". Due to the added support for WS-Security this is now an unnecessary restriction.

Removal of 'Fields' element

The agent action element part of the message e.g. "SetIndicator", in version 1.0 required the action fields to be wrapped in an outer Fields element. Due to improvements in the underlying web service standards supported (e.g. JAXB) this is now not required resulting in a smaller message size.

13. Customs Info Integration

The Global Trade Management component of Oracle Transportation Management supports an interface to Customs Info for retrieving reference data used in the services. This section details the setup for accessing the data.

Customs Info Registration

Registration with Customs Info is required for access to the Customs Info data for Global Trade Management. The registration process will give a site ID and password to the client that can be entered in the setup of Global Trade Management. This user and password combination will be used to download the actual data.

Setup in Global Trade Management

Global Trade Management Properties Setup

Network Connectivity to Customs Info Server

If the network Global Trade Management is installed requires a proxy server for external access to the Customs Info server, specify the following properties in the glog.properties:

```
glog.integration.http.proxyHost=proxy-address
glog.integration.http.proxyPort=proxy-port
```

Refer to the online help for additional details.

Properties for Data Loading Directories

The data loader relies on pre-defined directories for the source and output directories. The directories are defined via properties:

```
gtm.dataload.basedir=$temp.dir$/dataload
gtm.dataload.inputdir=$gtm.dataload.basedir$/input
gtm.dataload.outputdir=$gtm.dataload.basedir$/output
gtm.dataload.workingdir=$gtm.dataload.basedir$/output
```

These properties do not need to be changed for implementation. The logic will create the directories needed if they are not already created.

Setup of Customs Info User Credentials

After retrieval of the user credentials from Customs Info, configure the credentials in Global Trade Management as follows:

- Setup an External System record in Global Trade Management with the following details:
  - **User Name**: User ID provided during registration
  - **Password**: Password provided during registration
  - **Password (Confirm)**: Same value for password
  - **URL (in For HTTP/HTTPS section)**: URL for accessing the Customs Info status document. The URL may resemble the following: https://gtm.customsinfo.com/status/statusdocument.xml
- Update the Content Source to refer to the newly created External System record as follows:

- Access menu: Trade Master Data > Power Data > Data Loading > Content Source
- Edit the Content Source with ID: "CUSTOMS INFO" which has been predefined for Customs Info
- Add an entry in the Content Source Config section at the bottom as follows:
  - **ID**: Specify an Id (e.g. CI_GTM_DATA)
  - **Comm Method**: Specify "HTTPPOST"
  - **External System**: Select the external system created above
  - **Active**: Select to enable the config.

Processing Global Trade Management Content

The actions to initiate the Global Trade Management content loading are defined with process control. This is accessible from the following menu:

Trade Master Data -> Process Management

The following processes are available:

- **Download Data Content**: Used to retrieve the content from a Global Trade Management content source. For Customs Info, this process will download the latest files from the Customs Info server into the local Global Trade Management server.
- **Purge Data Content**: This is used to cleanup up the files on the Global Trade Management server used for the data loading.

1.

14. Global Trade Management Screening Service

This section captures the screening services developed in Global Trade Management.

All interactions with the Global Trade Management Services through integration is made using the Service Request element (refer to the GLogXML-GTM.xsd schema for additional details). Upon receiving the ServiceRequest XML in Global Trade Management, the request is processed synchronously and a ServiceResponse xml is generated as a response to the caller. Note that the Transmission/TransmissionHeader/TransmissionType must be set to "SERVICE" for the synchronous processing to occur.

The ServiceRequest element contains several options for the screening services. A few of the available services are as follows:

ElementName (Service)	Description
RestrictedParty	Used to indicate if the specified party is on the restricted parties list.
SanctionedTerritory	Used for transactional screening of territories. The screening uses the rules engine and runs all rules that have a control category of SANCTION. Users specify any number of territories and qualifiers in the LocationInfo element.
Classification	Used for transactional searching for global classification based on product description.
ComplianceRule	Used for transactional screening of personal information, product and usage information, location information, and user defined conditions.
LicenseDetermination	Used to find matching licenses for a license control, based on party, location and user defined code data.

Refer to the GLogXML-GTM.xsd schema file for additional fields in the ServiceRequest and ServiceResponse, and the online help for details on the services.

