Oracle® Retail Fiscal Management/RMS Brazil Localization

Implementation Guide

Release 13.2

January 2011

ORACLE®

Oracle Retail Fiscal Management/RMS Brazil Localization Implementation Guide, Release 13.2

Primary Author: Nathan Young

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.

(iii) the software component known as **Access Via™** licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(iv) the software component known as **Adobe Flex™** licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

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

Contents

9 RMS Advanced Queueing Implementation

10 Batch Processes

Preface

The *Oracle Retail Fiscal Management Implementation Guide* provides detailed information that is important when implementing ORFM.

Audience

This document is intended for the Oracle Retail Fiscal Management application integrators and implementation staff, as well as the retailer's IT personnel.

Related Documents

For more information, see the following documents in the Oracle Retail Fiscal Management Release 13.2 documentation set:

- *Oracle Retail Merchandising System Release Notes*
- *Oracle Retail Merchandising System Installation Guide*
- *Oracle Retail Merchandising System User Guide and Online Help*
- *Oracle Retail Sales Audit User Guide and Online Help*
- *Oracle Retail Trade Management User Guide and Online Help*
- *Oracle Retail Merchandising System Reports User Guide*
- *Oracle Retail Merchandising System Operations Guide*
- *Oracle Retail Merchandising System Data Model*
- *Oracle Retail Merchandising Batch Schedule*
- *Oracle Retail Merchandising Data Conversion Operations Guide*
- *Oracle Retail Merchandising Implementation Guide*
- *Oracle Retail Merchandising Licensing Information*
- *Oracle Retail Fiscal Management Data Model*
- *Oracle Retail Fiscal Management/RMS Brazil Localization Installation Guide*
- *Oracle Retail Fiscal Management User Guide and Online Help*
- *RMS/ReSA Brazil Localization User Guide and Online Help*

Customer Support

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

https://support.oracle.com

When contacting Customer Support, please provide the following:

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

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 13.2) or a later patch release (for example, 13.2.1). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Oracle Retail Documentation on the Oracle Technology Network

Documentation is packaged with each Oracle Retail product release. Oracle Retail product documentation is also available on the following Web site:

http://www.oracle.com/technology/documentation/oracle_retail.html

(Data Model documents are not available through Oracle Technology Network. These documents are packaged with released code, or you can obtain them through My Oracle Support.)

Documentation should be available on this Web site within a month after a product release.

Conventions

The following text conventions are used in this document:

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

1

Oracle Retail Fiscal Management Overview

What Is Oracle Retail Fiscal Management?

Oracle Retail Fiscal Management (ORFM) manages Nota Fiscal processing through integration with Oracle Retail Merchandising System (RMS), Oracle Retail Warehouse Management System (RWMS), and Oracle Retail Store Inventory Management (SIM).

In Brazil, before shipping any inventory out of the warehouse or store, it is mandatory to generate a Nota Fiscal to accompany the inventory movement. Similarly, prior to receiving physical inventory in a warehouse or store, it is mandatory to match the Nota Fiscal.

ORFM is integrated with RMS, RWMS, and SIM. Since ORFM and RMS share the same instance and database schema, ORFM can look up the RMS database tables and vice versa. For example, a purchase order (PO) or transfer created in RMS is directly accessible to ORFM while a supplier, location, or item level fiscal attribute that is set in RMS can be accessed by ORFM through direct database lookup. ORFM also integrates with SIM and RWMS using the Oracle Retail Integration Bus (RIB).

Functional Assumptions

ORFM/RMS Brazil localization has the following functional assumptions for the 13.2 release:

- AUTO_RCV_STORE Indicator on the supplier screen in RMS is never checked.
- Location-level security is not enabled in ORFM.
- For EDI NF, if there are observations related to recoverable ICMS and ICMS-ST, they should appear as different fields. The system cannot parse the observations.
- During discrepancy identification and resolution, the system always assumes the cost components on the NF are correct.
- During discrepancy identification, the system compares discounted cost on the purchase order and discounted cost on the NF. Unit cost (nondiscounted) and the discount values are not compared separately.
- The open purchase order quantity is affected only after physical receiving and not after fiscal receiving.
- For triangulation purchase order flow, the NFs from both the suppliers should match in cost and quantity and should be entered in the system at the same time.

- The taxes on the main NF and the delivery NF (complementary NF for triangulation) are mutually exclusive.
- The system generates correction documents for both main and delivery suppliers.
- If the purchase NF has taxes such as IPI and ICMS-ST that are added to the NF total, these taxes are embedded in the RTV cost on the RTV NF.
- Freight or any other cost component is not reversed on an RTV NF.
- The following fields in the NFe data mapping are out of scope:
  - CT-e is the electronic freight NF which is out of scope of this first release of NFE
  - Fiscal Coupon
  - NF-e Identification- Field #21-DANFE Print format
  - Collection Place
  - Delivery place
  - Imports Declaration
  - Additions
  - B2B
  - New vehicles Detailing
  - Medicine Detailing
  - Weapons Detailing
  - Fuel Detailing
  - IPI
  - IPI – CST 00, 49, 50, 99
  - Importation Taxes
- For triangulation purchase order flow, both the delivery and main supplier should be known when the purchase order is created.
- If the stock being returned from a location was received at that location through a transfer, the system cannot track the last purchase NF and hence the RTV happens at WAC.

What Is the Retail Taxation Integration Layer?

The Retail Taxation Integration Layer (RTIL) is implemented as a Java Enterprise Edition application, hosting the canonical Web services and the associated tax service provider adapters. This layer forms the conduit between the Oracle Retail applications and the tax service provider. The Retail Taxation Integration Layer is responsible for the assembling and disassembling of the vendor specific data model to a canonical tax data model based on the configuration. The RTIL hosts vendor-specific connectors which can communicate to the external third-party services with various protocols such as EJB, JMS, SOAP, and HTTP, based on the configuration. The subscribing application should have a vendor-neutral canonical tax rules (CTR) Web service callout utility for making Web service calls. The subscribing application is not aware of the tax service provider. RTIL acts as a bridge between the subscribing application and third-party tax service provider.

What Is the Mastersaf Tax Engine?

The Mastersaf Tax Engine aids retailers with multiple state operation, with a high level of complexity and large number of transactions, items and locations.

ORFM is integrated with the Mastersaf Tax Rules (MTR) to address all of the Brazilian tax legislation with a high level of exception treatments. For all flows in Oracle Retail that need to have tax calculations, Mastersaf Tax Rules verifies that all taxes are applied considering the input parameters.

The following processes in ORFM/RMS use the tax calculation integration:

- Inbound Nota Fiscal validation
- Outbound Nota Fiscal issuing
- PO tax breakdown
- Item creation

What Is Nota Fiscal?

In Brazil, all movements of products from one location to another, or from a supplier to the retailer's location and vice versa, must be accompanied by a fiscal document called a Nota Fiscal (NF). This document contains all the information related to the transaction. It can be compared to a bill of lading (BOL) because it has the items and quantities. NF also has financial information such as the cost, so it can be compared to the invoice too. In addition, this document has all the taxation and fiscal information, which make the NF a unique document with all the related information. The management and processing of this document is closely linked to the business process flows of receiving, shipping, and all types of transactions that affect the inventory.

What Is Nota Fiscal Eletronica (NF-e)?

NF-e is a Brazilian government project tasked with implementing a national model of electronic fiscal documentation to replace the current system of issuing the fiscal documents in paper. The virtual document will have juridical validity guaranteed by the digital signature of the issuer. It will simplify the fiscal obligations of the taxpayers and will allow the follow-up of the commercial operations by the tax authority.

The NF-e issuer will generate an electronic file with all NF information in a more detailed level than the regular NF. This file must be digitally signed to guarantee the integrity of the data and the authorship of the issuer. This electronic file that corresponds to the Nota Fiscal Eletrônica (NF-e) is transmitted by the internet to the Secretaria da Fazenda - Brazilian Tax Authority (SEFAZ) of the origin state of the issuer. The SEFAZ provides a pre-validation of the file and returns a receiving protocol (Authorization for Use), that will be necessary to the traffic of the goods.

To follow the goods, a graphic representation of the NF-e will be printed. The DANFE (Documento Auxiliar da Nota Fiscal Eletrônica - Auxiliary Document of the Electronic Invoice) will be printed in a common paper, one copy that will have highlighted the access key for consultation of the NF-e in the internet, and a bidimensional bar code which will facilitate the capture and confirmation of information of the NF-e by the fiscal units.

The DANFE is not a Nota Fiscal, and does not replace the NF. It is an auxiliary document for consultation of the NF-e. It has the access code of the NF-e, which allows its owner to confirm the real existence of the NF-e in the Receita Federal Brasileira - Brazilian Federal Tax Authority (RFB) environment or the SEFAZ Web site.

2

RMS Setup for ORFM and Brazil Localization

This chapter describes the processes for preparing RMS to use ORFM and Brazil localization.

Decouple RMS and ORFM Batch

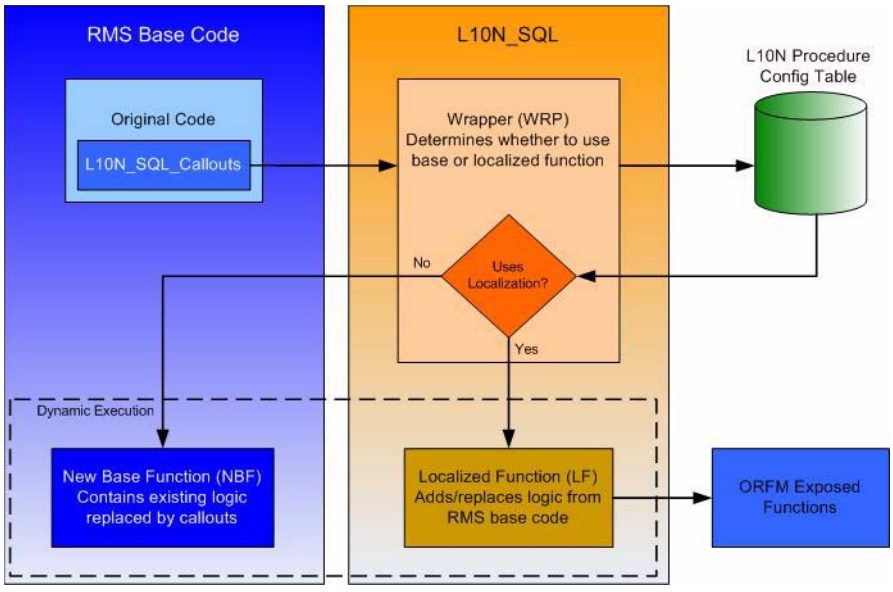
Identical processes that are performed differently in RMS and ORFM have been separated using a process called decoupling. This allows installation of RMS without installing any (or a minimal portion) of the ORFM modules. This means that direct reads and writes to the ORFM tables and views from RMS are avoided.

All processes involving localization are encapsulated within a localization wrapper package. The decoupling between RMS and ORFM is controlled in that wrapper package.

Dynamic Function Calls

ORFM and RMS utilize dynamic function calls to decouple. This method uses a configuration table to determine which function or procedure, either base or localized code, is dynamically executed.

Figure 2–1 Dynamically Executed Decoupling



There are five components in the dynamic call approach:

- **The original code -** This is the base code that does the callout to the localization layer.
- **L10N_SQL wrapper (WRP)** - This function decides whether to use the base code or the localized version of the code.
- **L10N configuration table** - This table contains the procedure name to be executed per localized instance.
- **New base function (NBF) -** This function contains the existing logic from the original code replaced by callouts to the localization layer.
- **Localized function (LF) -** This contains the necessary procedures to carry out the localization tasks. This may contain actual calls to ORFM functions.

PL/SQL Package Calls

Callouts are made in the original base code where localized codes are introduced. There can be more than one callout in a base function, and every callout points to the localization layer.

In scenarios in which code needs to be replaced, or the base code block is not applicable when localization is enabled, a New Base Function (NBF) exists to contain this code. The localized functionality is moved to a Localized Function (LF).

The localization layer contains the WRAPPER function (WRP) - EXEC_FUNCTION, which is called by the original base code. This wrapper uses a L10N configuration table to determine if the New Base Function (NBF) or a Localized Function (LF) needs to be executed. The call to these functions is done dynamically. The configuration table may contain different function references for different localizations.

Because the calls to the functions are dynamic and the parameters passed in and out may vary depending on the function called, the PL/SQL Type Inheritance approach is used to accomplish this.

PL/SQL Type Inheritance

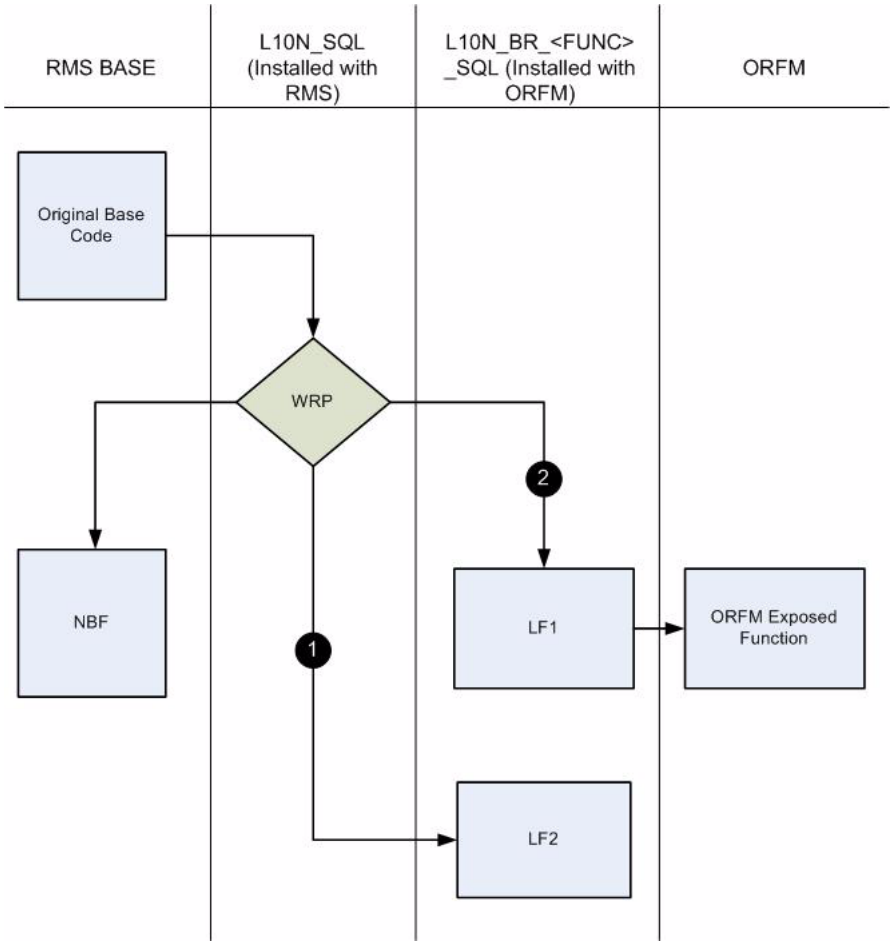
In this approach, the in/out parameters to be passed to functions should be encapsulated in object types. The object types follow PL/SQL Type Inheritance (a supertype is created from which several subtypes will be derived). The supertype is a generic object that can be used across functions. Derived subtypes contain all the attributes of the parent type (or supertype). The subtypes can also contain additional attributes.

Localization Layer

The localization layer is made up of the following components:

- The L10N configuration table
- The L10N_SQL package
- Country-specific localization packages (L10N_<CN>_SQL)

Figure 2–2 Localization Layer



L10N_SQL contains only the wrapper and other functions, which are generic to localized countries.

L10N_BR_<FUNC>_SQL contains only the Brazil-specific functions under <FUNC> functionality area that are called by the L10N wrapper. These functions may (flow line (2) in the diagram) or may not (flow line (1) in the diagram) call ORFM functions directly.

Accomplishing Decoupling

BASE INSTALL: RMS has a dependency on L10N_SQL because it calls the wrapper directly through a callout. When RMS is installed, an L10N configuration table is prepopulated with references to the New Base Functions (NBF). Because these NBF functions have no dependency on ORFM, RMS does not need the Localized Functions (LF) in the base.

LOCALIZATION INSTALL: After ORFM and country-specific localization packs (L10N_BR_<FUNC>_SQL, L10N_PE_<FUNC>_SQL) are installed, the L10N configuration table is prepopulated with the localized functions for the country (BR, PE). The wrapper now calls the functions from the country-specific packages.

Note: Dependency is created only at run time because the functions are executed dynamically.

Decoupling Batch Programs

In a batch program, decoupling is accomplished in a way that, if a base and localized process must occur (forking), the base logic will reside in the NBF andthe localized process will be in the LF. A wrapper is used by the main batch program to call the LF or NBF. Only one version of the function is executed in an instance of the function call.

To decouple the functions, there are separate localized libraries for LFs per country. NBFs are declared in the main batch program.

A utility library contains all the utility functions including the wrapper function. The utility function is used by the main batch program to find and retrieve the LF or NBF it has to execute. The LF pointers are organized based on the batch program that uses them, so one batch program can only use its set of LF pointers.

The NBF pointers are declared in the main batch program.

To make the parameters of the NBFs and LFs generic, a struct is used. This struct, which is called the parent struct, resides in an object library. This is the only parameter for the NBF and LFs. If additional fields are needed, the library must be modified to create a struct within the parent struct (the child struct). These structs must be populated first before passing it to the wrapper class to call the NBFs or LFs.

Set Up Fiscal Download

Fiscal Download holds data such as NOP, ncm_codes, pauta_codes and NCM exception codes. This data is obtained from Mastersaf and is maintained in RMS. This is mandatory information for Item and NF creation for the tax retrieval from Mastersaf.

No front-end navigation is available for this. There is a batch available to download the data.

Set Up Item Attributes

Item attributes are mandatory for any item creation. These attributes are referenced when the item is used for any kind of transaction data creation like Purchase Orders and Transfers. Item attributes include NCM code, service ind (Y/N), NCM characteristic code, ex ipi, pauta code, service code, origin code, and federal service code.

Navigate: From the main menu, select Items > Country > Fiscal Attributes.

Set Up Fiscal Attributes

Fiscal attributes hold information such as address, company code, and tax contributor indicators. These are necessary for tax calls and are used by MTR to apply appropriate rules and return the correct taxes.

Navigation is available for the setting up fiscal attributes. This is available from the Options menu of Warehouses, Stores, Outside Locations, Partners, and Company

Set Up Utilization

All merchandise that is being fiscally accepted using ORFM must have the fiscal utilization, which determines the type of the business operation that the fiscal document contains.

The fiscal utilization determines the appropriate taxes involved by the retail operation, the impact on stocks and costs, and the type of information to be sent to other systems.

Utilization is associated with a Requisition Type and a Nature of Operation. This identifies the type of transaction for MTR to calculate appropriate taxes.

Utilization also can be associated with a particular Document Type. Different attributes are available to be configured for utilization which gives more granular information on the kind of document and action to be applied. The various parameters are:

- Complementary NF for Triangulation
- Complementary NF for Freight
- ICMS-ST recovery
- Allow Receiving
- Automatic NF Approval
- Choose NF

Navigate: From the main menu, select Fiscal Management > Fiscal Configuration > Fiscal Utilization > Edit. The Fiscal Utilization Setup window opens.

Set Up Document Types

You must set up at least one fiscal document type prior to using ORFM. Each Fiscal Document Type needs to be associated with a fiscal utilization. Multiple Utilization Ids can be associated to one document type.

Navigate: From the main menu, select Fiscal Management > Fiscal Configuration > Fiscal Document Types > Edit. The Fiscal Document Type window opens.

Set Up System Options

ORFM System Parameters, including the Default Utilization codes need to be set up. The configuration settings control system behavior based on the values entered.

Navigate: From the main menu, select Fiscal Management > System Setup > System Options > Edit. The System Options window opens.

Set Up Reason Codes

You must set up reason codes for overage and damaged in the RMS reason code master table (inv_adj_reason) prior to using ORFM. These reason codes are interfaced to ORFM from RWMS and SIM. If these reason codes are not set up correctly, the integration to ORFM for receipt verification will fail.

Set Primary Country

To use ORFM, the Primary Country must be set to **BR**. This can be done during the RMS installation process. See the *Oracle Retail Merchandising System Installation Guide* for more information.

Set VAT Indicator

To use ORFM, VAT_IND must be set to **Y**. This can be done during the RMS installation process. See the *Oracle Retail Merchandising System Installation Guide* for more information.

Set Default Tax Type

To use ORFM, the Default Tax Type must be set to **GTAX**. This can be done during the RMS installation process. See the *Oracle Retail Merchandising System Installation Guide* for more information.

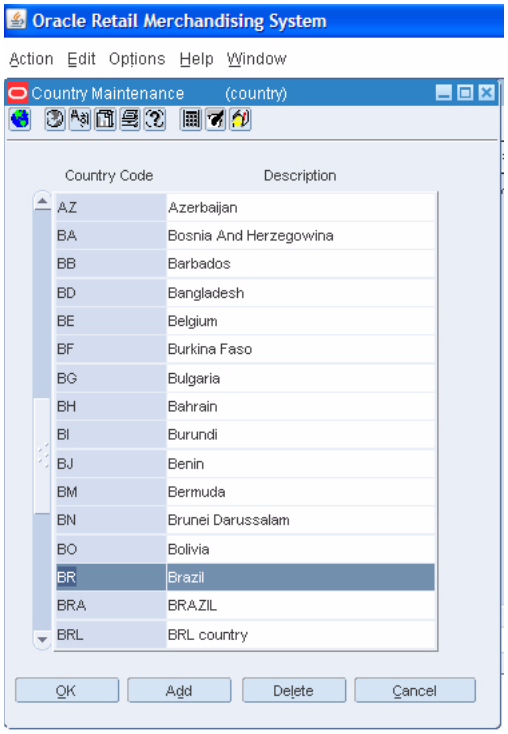
Verify Localized Indicator

The Localized indicator check box indicates whether the Brazil patch is installed. It is auto-checked by the system if the Brazil patch is installed.

Use the following procedure to verify the Localized Indicator:

Navigate: From the main menu, select Control > Setup > Country > Edit. The Country Maintenance window opens.

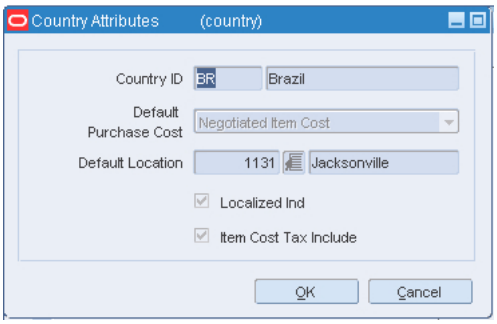
Figure 2–3 Country Maintenance Window



1. Select BR and click the Options menu.

2. From the Options menu, select Attributes. The Country Attributes window opens.

Figure 2–4 Country Attributes Window



3. Verify that the Localized Ind is checked and click **OK**. You are returned to the Country Maintenance window.
4. Click **OK** to save your changes and close the window.

Set Up Tax Codes

Tax codes must be set up in the FM_TAX_CODES table. This table holds entries for all possible tax codes that can appear on a Nota Fiscal which would be applicable on any given transaction. It also holds the matching_ind (Y/N) identifier which drives the tax discrepancy identification functionality of ORFM.

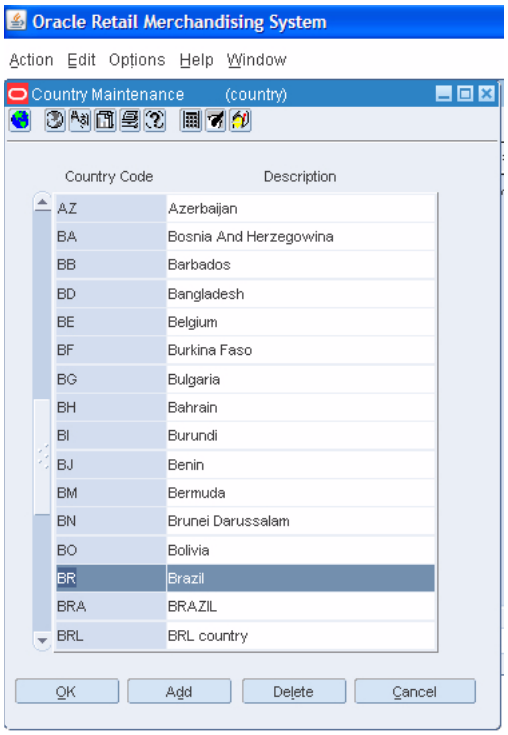
This table is populated from the back end and does not have any navigation available.

Set Up Flex Field Attribute

The fiscal code of the country must be set up before using ORFM.

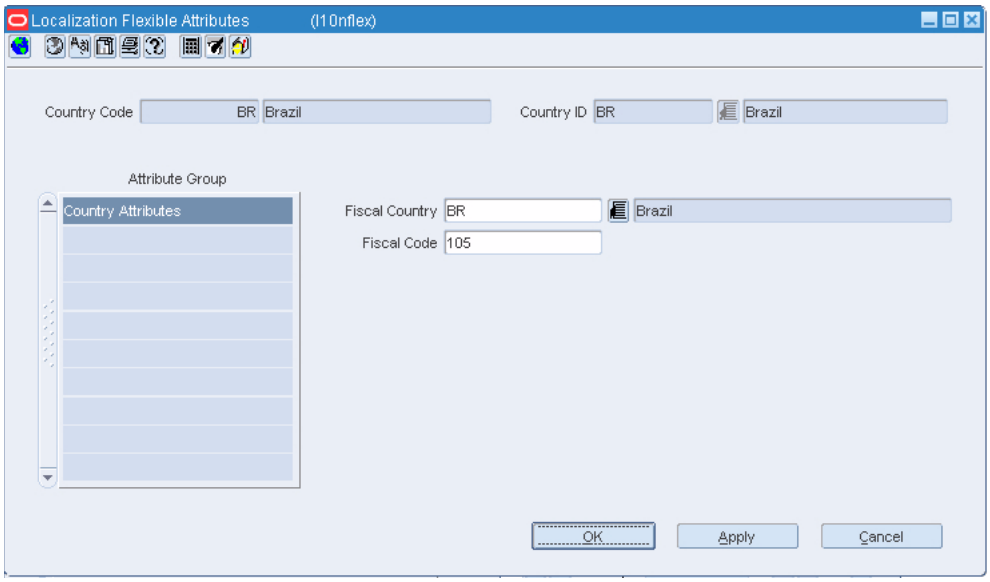
Navigate: From the main menu, select Control > Setup > Country > Edit. The Country Maintenance window opens.

Figure 2–5 Country Maintenance Window



1. Select BR and click the Options menu.
2. From the Options menu, select Fiscal Attributes. The Localization Flexible Attributes window opens.

Figure 2–6 Localization Flexible Attributes Window



3. Set the Fiscal Code to 105 and click **OK**. You are returned to the Country Maintenance window.
4. Click **OK** to save your changes and close the window.

3

ORFM Integration with Other Applications

This chapter provides a functional overview of how ORFM integrates with other systems.

Overview

The first section in this chapter provides a diagram illustrating the various Oracle Retail products and databases with which ORFM interfaces, as well as the overall dataflow among the products. The accompanying explanations are written from a system-to-system perspective, illustrating the movement of data.

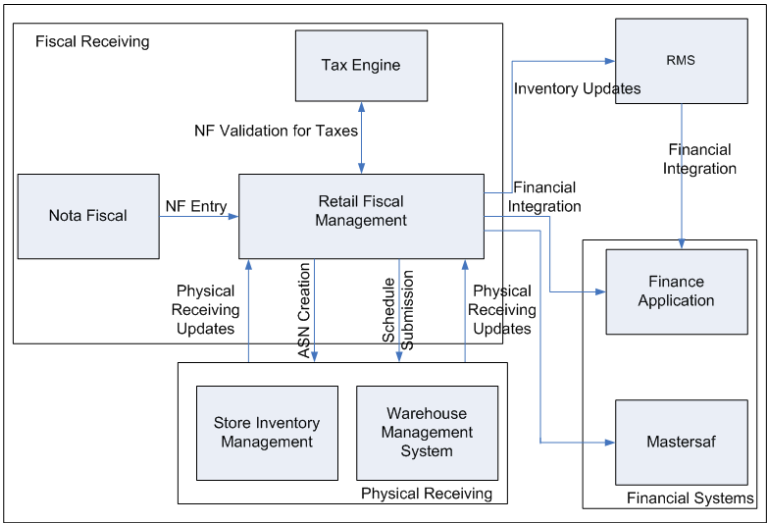
The Nota Fiscal receiving (inbound) and issuing (outbound) are controlled within ORFM and all integration with RMS, RWMS, and SIM is based on the physical movement of the products.

Nota Fiscal Receiving

When ORFM is enabled, SIM and RWMS are not allowed to physically receive any inventory prior to fiscal receiving (that is, inbound Nota Fiscal processing) is completed in ORFM. After physical receiving in SIM or RWMS, the inventory in RMS is not updated until the fiscal receipt and physical receipt comparison is completed in ORFM and any discrepancy is resolved.

The following diagram outlines the ORFM business process for inbound operations:

Figure 3–1 Inbound Process Flow



Inbound Process Flow

The inbound process flow is as follows:

1. Enter the Nota Fiscal (NF) in the Retail Fiscal Management module.
2. When the shipment arrives at the warehouse or the store, create a schedule and enter the NFs received.

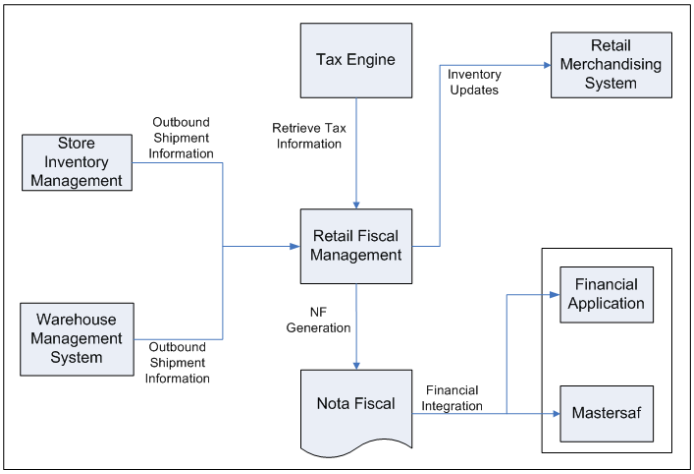
> **Note:** More than one NF can be linked to a schedule.

3. After NF entry, validate the NFs. In the validation process, the application checks for data integrity. Match the NF with the requisition documents in Retail Merchandising System (RMS). This process is called as Fiscal Receiving.
4. If the NF and the PO does not match, the NF is in discrepancy. You can identify the following discrepancies with the ORFM application:
   - Unit Cost Discrepancy for each item in the NF
   - Quantity Discrepancy for each item in the NF
   - Tax Discrepancy at aggregate level (such as NF header level or individual item level) for all the applicable taxes
5. After validation, send the schedule to the Warehouse Management System (RWMS), and Store Inventory Management (SIM). After physical receiving, RWMS and SIM publish the receipt updates to ORFM. This completes the NF processing.
6. After NF processing is complete, ORFM calls RMS to update inventory and WAC.
7. Send the transaction data in RMS and ORFM to a financial application. For fiscal reporting purposes, send the NF data in ORFM to the fiscal reporting system like Mastersaf.

Because the Nota Fiscal contains taxation information, ORFM calls the tax engine for calculations during the receiving and issuing of a Nota Fiscal.

Nota Fiscal Issuing

When ORFM is enabled, RMS inventory is not updated immediately when any inventory is shipped out from RWMS or SIM. The inventory movement information is first sent to ORFM for the generation of the outbound Nota Fiscal. After approval of the outbound Nota Fiscal, ORFM sends the inventory updates to RMS.

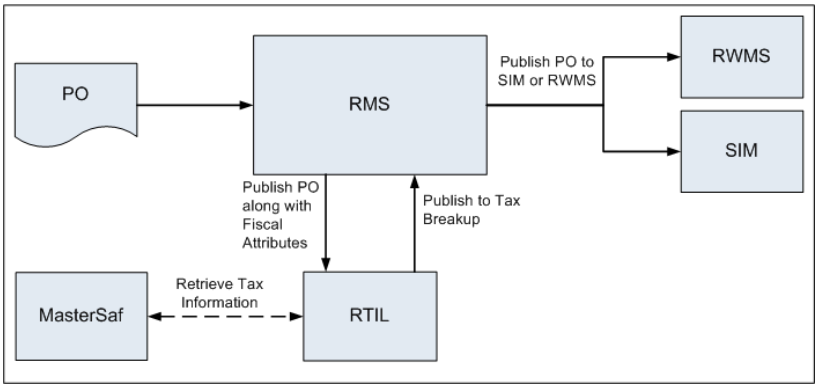


Outbound Process Flow

1. Enter the outbound shipment information in SIM or RWMS and send it to ORFM.
2. ORFM sends the outbound shipment information to the tax engine. The tax engine figures the tax information and returns it to ORFM.
3. Create the Nota Fiscal based on the outbound shipment information and the tax information.
4. After NF processing is complete, ORFM calls RMS to update inventory and WAC.

Purchase Orders

In Brazil, receiving without a valid purchase order (PO) number is not allowed, because fiscal receiving must precede physical receiving. For this reason, a direct store delivery (DSD) without a purchase order cannot be received in SIM. A purchase order created in RMS should be sent to RWMS and SIM through RIB as per the base process flow. The purchase order created should flow to the Mastersaf Tax Engine through the Retail Taxation Integration Layer (RTIL) along with the fiscal attributes of the supplier(s), receiving location(s), and the items. The Mastersaf Tax Engine should calculate the tax breakup and send it back to RMS through RTIL.



1. Create a purchase order in RMS.
2. Publish the PO. Retrieve the Brazil tax information and include it in the PO. Send the PO with the Brazil tax information to RTIL.
3. RTIL publishes the tax breakdown of the PO.
4. The PO is published in SIM or RMS.

ORFM has direct access to the PO created in RMS and the related fiscal details. This is required to validate and match the Nota Fiscal. The following PO details are relevant to ORFM:

- PO header information
- Order supplier master data
- Order supplier site master data
- Order supplier site fiscal attributes
- Delivery supplier master data
- Delivery supplier site master data
- Delivery supplier site fiscal attributes
- Destination locations master data
- Destination location fiscal attributes
- PO detail information
- Non-merch cost details
- Item master data
- Item fiscal attributes

Triangulation

ORFM supports the triangulation process. Triangulation is a typical purchase transaction where the PO is raised for a supplier (referred to as main/ordered supplier) but the goods are shipped to the retailer by a different supplier (referred to as the delivery supplier). In Brazil, for triangulation transactions, NFs are received from both the main and delivery suppliers. Both the NFs are for the same stock but the details on both the NFs may vary. For example, federal tax such as IPI usually appears on the main supplier's NF, while state taxes such as ICMS and ICMS-ST usually appear on the delivery supplier's NF.

ORFM and RWMS

The Brazil requirements are strict for receiving exactly what is on the NF. ORFM provides delivery information to RWMS, from which the user creates receiving appointments. The ability to alter appointment quantities is not available based on system parameter configuration. Validation exists to compare the appointment quantities with the quantities supplied by ORFM. Users can split items as needed by altering case pack sizes, but the totals must match with ORFM.

Requirements for each appointment are grouped into the same unique schedule number provided by ORFM. That schedule number is added to the appointment table and is searchable by users.

ORFM and SIM

Nota Fiscal (NF) is a Brazilian fiscal legal document that needs to be generated to follow the physical movement of goods. ORFM handles the specific processing.

SIM is impacted by NF in the following ways:

- Shipping to warehouse or vendor
- DSD Receiving
- Transfer receiving
- Inventory Adjustments and stock counts
- Receive Unit Adjustments

The receiving process in SIM for internal (warehouse and store) deliveries does not allow under receiving or over receiving because of the following:

- Transfers cannot be modified in Brazil through the user interface security; they are auto-received.
- No adjustments are allowed in Brazil.

For warehouse deliveries, NF and SIM get the ASN information from the warehouse. After the ASN has been processed in NF, SIM gets a second ASNin message. This message launches a new auto-receive process. Security is used to limit users' access to the warehouse delivery dialog.

The reason for SIM to still subscribe to the ASNin message from RWMS or another store is so that the system knows which items are on their way while they are in transit, for more accurate inventory information.

ORFM Integration with a Financial Application

Note: There is no integration available between ORFM and any financial application. All the NF transaction data is posted into financial tables in ORFM.

Transaction data in ORFM maintains NF information so that it can be sent to the financial application.

Different transaction codes are used for tax-exclusive cost (also referred to as BC), cost components (such as freight or insurance), taxes that are input creditable, and taxes that affect cost (non-input credit type).

For tax-related transaction data, the user can configure the system to make a separate entry for each tax code. By doing so, the user can map the same transaction code having different tax codes in different accounts.

4

ORFM Integration – Functional

ORFM is integrated with other Oracle Retail applications including RMS, RWMS and SIM. Because RMS and ORFM share the same instance and same database schema, ORFM can look up the RMS database tables and vice versa. A purchase order (PO) or transfer created in RMS is directly accessible to ORFM. Supplier, location or item level fiscal attributes that are set in RMS can be accessed by ORFM through direct database look up.

ORFM integrates with SIM and RWMS through RIB (for real-time integration). If Brazil localization is enabled, SIM or RWMS should not be allowed to physically receive any inventory before fiscal receiving (inbound NF Processing) is completed in ORFM. After physical receiving in SIM or RWMS, the inventory in RMS should not be updated until fiscal receipt and physical receipt comparison is done in ORFM and the discrepancy (if any) is resolved as explained above. See "Overall Solution Landscape," which depicts the inbound operations process flow.

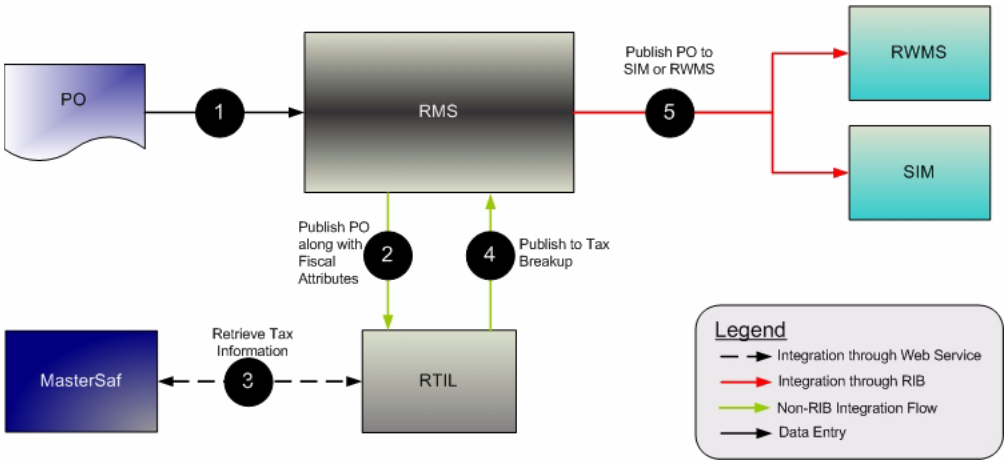
If Brazil localization is enabled, RMS inventory is not updated immediately when any inventory is shipped out from RWMS or SIM. The inventory movement information should be sent to ORFM for the generation of outbound NF. After approval of the outbound NF, ORFM should send the inventory updates to RMS. The Outbound Operations Process Flow diagram in the "Overall Solution Landscape" section depicts this.

In the subsections below, each transaction with a different integration flow (other than the base application) for Brazil is detailed.

Purchase Order Receiving

In Brazil, receiving without a valid PO number is not allowed because fiscal receiving must precede physical receiving. Therefore, a direct store delivery without a PO cannot be received in SIM. A PO created in RMS is sent to RWMS and SIM through RIB as per the base process flow. The PO created flows to the Mastersaf Tax Engine through the Retail Taxation Integration Layer (RTIL), along with the fiscal attributes of the suppliers, receiving locations, and the items. The Mastersaf Tax Engine calculates the tax breakdown and sends it back to RMS through RTIL.

Figure 4–1 Purchase Order Receiving



A purchase order goes through the following process be published to RWMS or SIM.

1. The purchase order is created in RMS.
2. The purchase order, along with the PO's fiscal attributes, are published to RTIL.
3. The Brazil tax information is retrieved and appended to the purchase order and sent back to RTIL.
4. RTIL publishes the tax breakdown of the purchase order.
5. The purchase order is published to SIM or RWMS.

> **Note:** Steps 2, 3 and 4 depicted above are specific to Brazil localization. Steps 1 and 5 are the same as in base RMS.

ORFM has direct access to the PO created in RMS and the related fiscal details. This is required to validate and match the Nota Fiscal. The following PO details are relevant to ORFM:

- PO header information
- Order supplier master data
- Order supplier site master data
- Order supplier site fiscal attributes
- Delivery supplier master data
- Delivery supplier site master data

- Delivery supplier site fiscal attributes
- Destination locations master data
- Destination location fiscal attributes
- PO detail information
- Non-merch cost details
- Item master data
- Item fiscal attributes

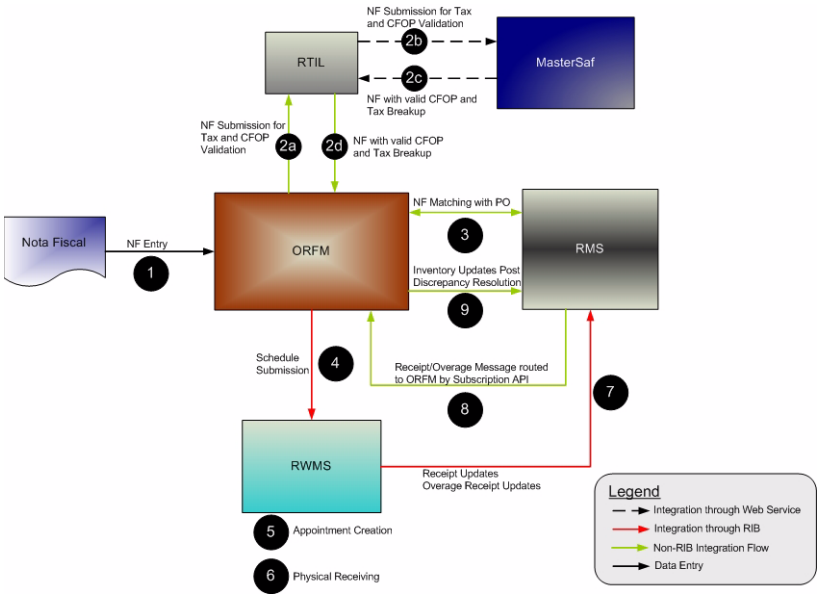
PO Receiving at a Warehouse

For a PO with a warehouse as the receiving location, the NF schedule status is Submitted. The message confirming that fiscal receiving is complete is published to RWMS through the Oracle Retail Integration Bus (RIB). The RIB message contains the following information:

- Schedule number
- PO number
- Receiving physical warehouse number
- Distinct item IDs per PO
- Consolidated quantities per item-PO
- UoM

One Nota Fiscal may contain items pertaining to several POs, and one schedule may contain several Nota Fiscal documents. It is likely that the supplier ships the same item on a PO through multiple NFs, and these NFs are part of the same schedule. If so, ORFM consolidates the item quantity by PO number before publishing it to RWMS.

Figure 4–2 PO Receiving at a Warehouse



After the automatic appointment creation, the RWMS user can schedule the appointment and proceed with the physical receiving. If Brazil localization is enabled in RWMS, receiving an item that is not in the appointment or receiving any extra quantity than in the appointment is restricted. Any such overage should be received separately in RWMS as an overage receipt.

After completing the receipt, as the appointment is closed, the receipt and the overage receipt are published by RWMS through RIB. RMS consumes these messages and the Subscription API routes the messages to ORFM (in Brazil Environment). When receiving, the same receipt message is used that is used to update receipt information to RMS. Though not all fields in the receipt message are relevant to ORFM, ORFM still consumes the entire message and persists it until the RMS inventory is updated.

The Receipt Overage message is a RIB message from RWMS to ORFM. The message contains the following information:

- Schedule Number
- ASN Number (Null for Warehouse receipts)
- Item Id
- Quantity Over-received
- UoM
- Reason Code

After ORFM consumes the Receipt and Receipt Overage messages, the discrepancy between physical and fiscal receiving is processed through the discrepancy resolution module. The correction letter and/or Return NF (for an overage receipt) are generated. No inventory updates happen in RMS for the overage receipt.

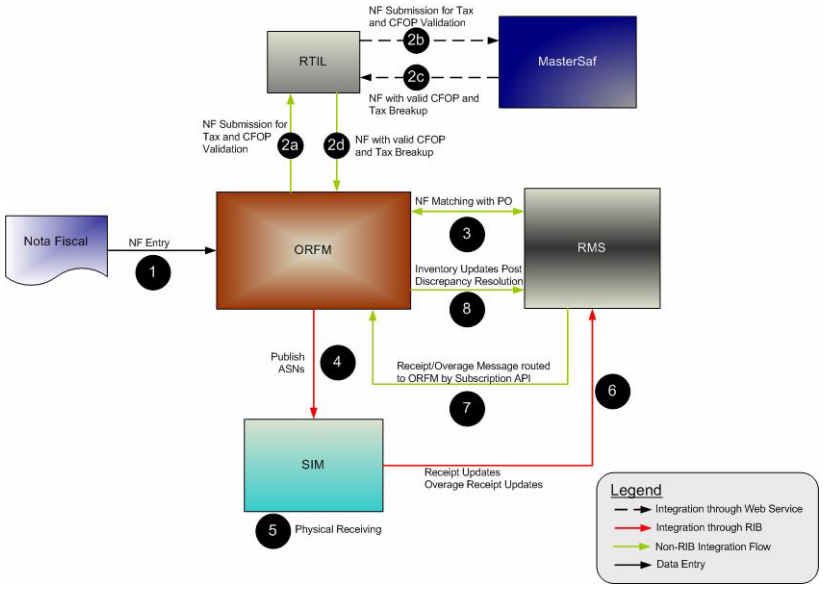
For the received items, the taxes are stripped from the unit cost to arrive at the tax-exclusive cost, additional cost components (if any) are added to calculate the actual landed cost, and the inventory and WAC updates are sent to RMS along with the NF number. Other than the cost and NF, all other information regarding the shipment is taken from the receipt message received from RWMS.

PO Receiving at a Store

For a PO with Receiving Location as a Store, if the NF schedule status is Submitted, the ASNs (ASNin) are generated and published to SIM. SIM then consumes this message to create the inbound ASN.

One NF can contain items pertaining to several POs, and one schedule can contain several NFs. It is likely that the supplier ships the same item on a PO through multiple NFs, and these NFs are part of the same schedule. If so, ORFM consolidates the item quantity by PO number before publishing it to SIM.

Figure 4–3 PO Receiving at Store



After the ASN message is received, the SIM user can proceed with the physical receiving. For Brazil localization, PO receiving without a valid inbound ASN is not permitted in SIM. In addition, receiving an item that is not in the ASN, or receiving any extra quantity than in the ASN, is restricted. SIM can enforce this restriction in two ways:

- The user can continue receiving inventory beyond the expected quantity on the NF, but the extra quantity will be removed from the transaction when confirming the delivery .
- The user is not allowed to receive any units above the expected quantity.

When receipt is completed for each ASN, the receipt and the overage receipt per ASN are published to RMS through RIB. For a receipt, the same receipt message is used that is used to update receipt information to RMS. The RMS subscription API does not consume the receipt message for the Brazil environment, but it is diverted to ORFM. ORFM stores the information in a staging table until all the ASNs in a schedule are received. The schedule status is changed to Received only when all the ASNs in a schedule are received. Referential integrity between the schedule number and the ASN is maintained in ORFM. SIM has no visibility to the schedule number. Not all the fields in the receipt message are relevant in ORFM, but ORFM still needs to consume the entire message and persist it until RMS inventory is updated.

The Receipt Overage message is a RIB message from SIM to ORFM.

After ORFM receives the Receipt and Receipt Overage messages, the discrepancy between physical and fiscal receiving is processed through the discrepancy resolution module. As required, the correction letter and/or Return NF (for an overage receipt) are generated. No information about the overage receipt flows to RMS from ORFM. For over-receipt, no inventory updates happen in RMS.

For the received items, the taxes are stripped off from the unit cost to arrive at the tax-exclusive cost. Additional cost components (if any) are added to calculate the actual landed cost. ORFM then calls RMS to update inventory and WAC. Other than the cost and the NF number, all other information regarding the shipment is taken from the receipt message that SIM has published.

Transfer Shipment and Receiving

A transfer (inter-company as well as intra-company) or allocation created in RMS is sent to RWMS and SIM through RIB.

ORFM has direct access to the transfer and the related fiscal details.

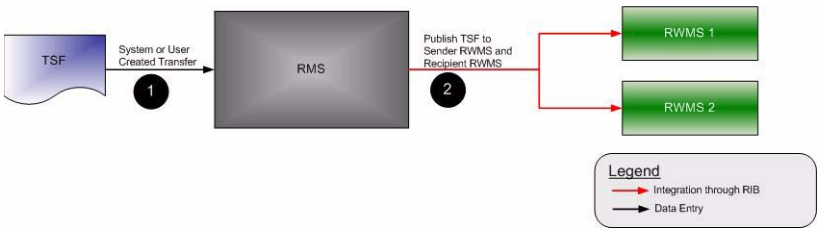
Transfers

Note: The following section uses warehouse-to-warehouse transfers to illustrate the transfer process. Other transfer processes, including warehouse-to-store and store-to-store are simlar from an ORFM perspective.

Transfer Creation

If a warehouse-to-warehouse transfer (or allocation) is initiated in RMS, the transfer request flows to RWMS (Sending Location and the Receiving Location) through RIB.

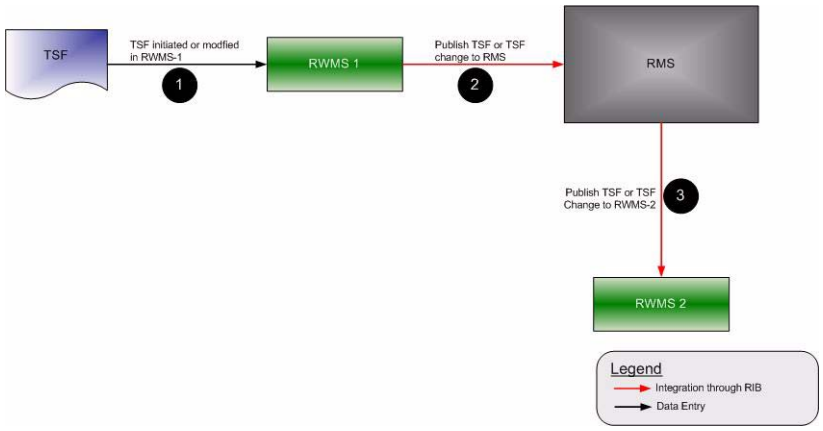
Figure 4–4 Warehouse to Warehouse Transfer Creation



Note: The sender warehouse and the recipient warehouse may run on the same instance or different instances of RWMS. If both warehouses run on the same instance of RWMS, RWMS-1 and RWMS-2 in the above image would be the same.

If a warehouse-to-warehouse transfer is initiated in RWMS, it flows to RMS and the other RWMS through RIB. If the RMS-initiated transfer is modified in RWMS-1, the modifications flow back to RMS and the other RWMS.

Figure 4–5 Warehouse to Warehouse Transfer

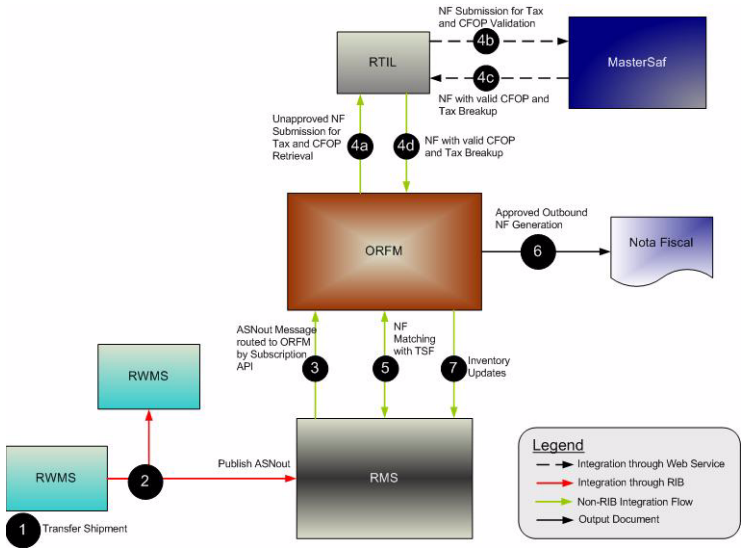


Transfer Shipment at Source

In Brazil, each inventory movement (shipment) should be accompanied by a fiscal document (NF). When the shipment goes out of the source warehouse, an NF is generated and approved. When RWMS ships the transfer out, ORFM is informed about the ASNout details so that the outbound NF can be generated. On approval of the outbound NF, the on-hand inventory of the source warehouse and in-transit inventory of the destination warehouse are updated in RMS.

The ASNout should be sent to the destination warehouse too, so that the destination warehouse generates the appointment and is prepared for the receiving.

Figure 4–6 Transfer Shipment at Source



When the transfer is picked and shipped in RWMS-1 at the source warehouse, ASNout is published and is subscribed to by RMS and RWMS of the destination warehouse simultaneously. In the Brazil environment, RMS routes the ASN message to ORFM. On consuming the ASN, ORFM generates the outbound schedules and outbound Nota Fiscal documents using the ASNout information. The tax information and the CFOPs are published on the NFs. NFs are validated before approval. On NF approval, the ASNout information is used to update the RMS inventory subsequently. The on-hand inventory of the source warehouse is decreased and the in-transit inventory for the destination warehouse is increased.

Transfer Receiving at Destination

When the shipment arrives at the destination warehouse, the user is expected to create an inbound schedule using the outbound schedule. There should be an inbound NF for each outbound NF in the schedule, and a one-to-one relationship must exist between the outbound schedule and inbound schedule, as well as outbound NFs and inbound NFs. As the schedule is entered and validated (retrieve appropriate inbound CFOPs through Mastersaf), the schedule can be submitted to RWMS-2 for receiving. Because RWMS-2 already has the ASN information, it is enough that the message published to RWMS-2 contains just the schedule number and the ASN numbers grouped under the same schedule. To avoid introducing a new RIB message, the schedule is submitted with the entire ASN (referred to as secondary ASN). RWMS can ignore the other information in the message and can just refer to the ASN numbers. The schedule number can be null, as in the case of publishing an ASN to SIM.

RWMS should create one appointment per schedule and combine all the ASNs for that schedule under the same appointment. After scheduling the appointment, the RWMS user can proceed with the physical receiving. It is possible that the user finds a discrepancy between the NF and the physical receipt. Any overage is consumed by ORFM, but no discrepancy resolution against overages is performed. In case of under-receipt, the receipt message coming from RWMS is the same as that of the ASN. In case of under-receipt in the destination warehouse , the inventory mismatch in the source warehouse should be handled manually by doing an inventory adjustment. In the Brazil environment, the RMS subscription API routes the receipt, overage, and inventory adjustment messages to ORFM. If the inventory adjustment reason code is associated with an utilization code, the NF should be generated in ORFM.

On receiving the receipt message from RWMS-2, the inventory updates are done in RMS. The in-transit inventory at the destination location is reduced and on-hand inventory is increased. WAC is recalculated for the destination location. Subsequent to the inventory updates for receipts, the inventory is updated for the inventory adjustment for any under-receipt.

After physical receiving is done at RWMS-2, RWMS-2 publishes the receipt and overage receipt (if any) message to ORFM through the RMS subscription API. Though ORFM may consume this message, ORFM does not use the information for updating the inventory. Any over-receipt at RWMS is handled outside the system.

Return to Vendor (RTV) Shipments

An RTV request can be initiated in RMS (for returns from store), or in SIM and RWMS. An RTV request initiated in RMS should flow to SIM through RIB as per the existing flow.

RTV Shipped from Warehouse

When the RTV is picked and shipped from the warehouse, an ASNout message should flow from RWMS to RMS through RIB. In the Brazil flow, after consuming the message, RMS routes it to ORFM; ORFM consumes the message from RMS.

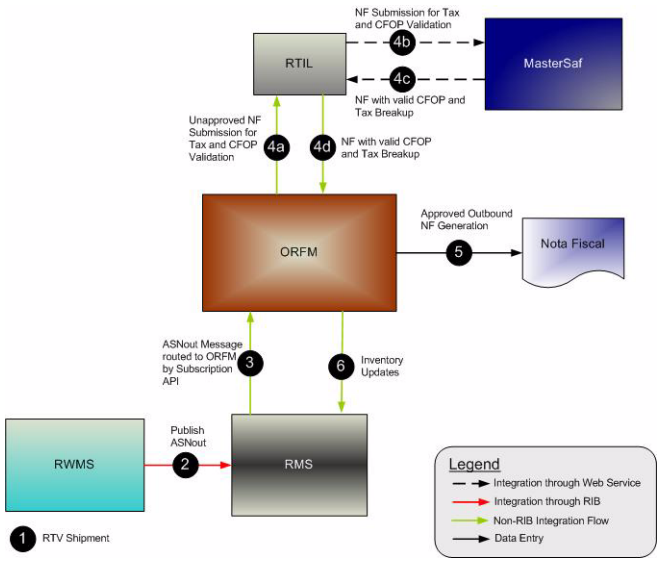
Using the information in ASNout message, ORFM should generate the outbound RTV NF. If there is reference to the inbound NF (through which the stock was received), the tax information is also retrieved from the inbound NF. If no inbound NF reference is required, the system should get the tax information through the Mastersaf Tax Engine. The Mastersaf Tax Engine should return appropriate CFOPs too. The NF should then be auto-approved. The shipment can leave the warehouse on printing of the approved NF.

The inventory updates are done in RMS and on-hand inventory is reduced. After the RMS updates are successful, the NF is approved. The RTV cost communicated to RMS is the tax-exclusive cost of the inventory returned. While making the transaction data entry, RMS should compare the RTV cost with the WAC, and if there is a mismatch, RMS should post a cost variance appropriately.

In Brazil, any outbound shipment should have a valid approved outbound NF accompanying it. Neither RWMS nor ORFM can control the outbound trailer leaving the warehouse or store without a valid NF. Business discipline must be followed to ensure this.

If for any reason the outbound NF cannot be generated or approved, the shipment cannot leave the source location. The RTV needs to be cancelled in that case. The system currently does not handle the cancellation of an RTV post-shipment from RWMS. It needs to be handled through manual inventory adjustments in RWMS and RMS simultaneously.

Figure 4–7 RTV Shipped from Warehouse



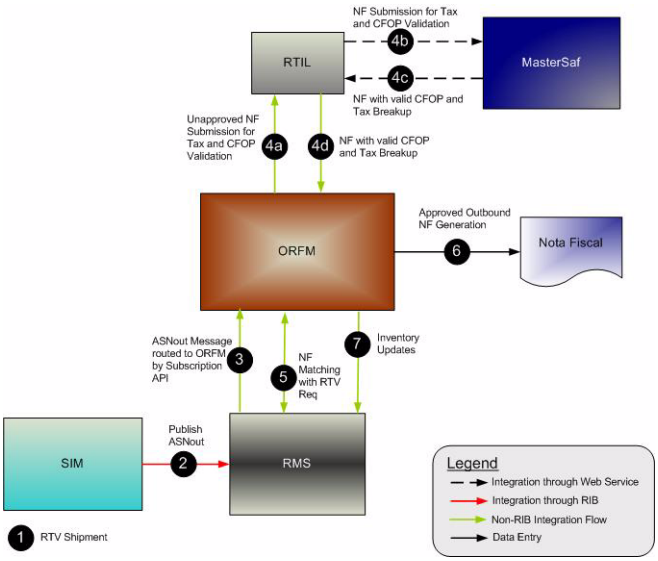
RTV Shipped from Store

From an ORFM perspective, RTV shipments from stores are similar to RTV shipments from RWMS. When the RTV is shipped from the store, the ASNout message flows from SIM to RMS through RIB. In a Brazil environment, RMS should direct the message to ORFM.

Using the information in the ASNout message, ORFM generates the outbound RTV NF, gets the tax information through the Mastersaf Tax Engine, matches the NF with the requisition (transfer request in RMS, if available), and approves the NF. The shipment can leave the warehouse on printing of the approved NF.

During NF approval, the inventory updates and transaction data posting API in RMS is called. After RMS updates are successful, the NF is approved.

Figure 4–8 RTV Shipped from Store



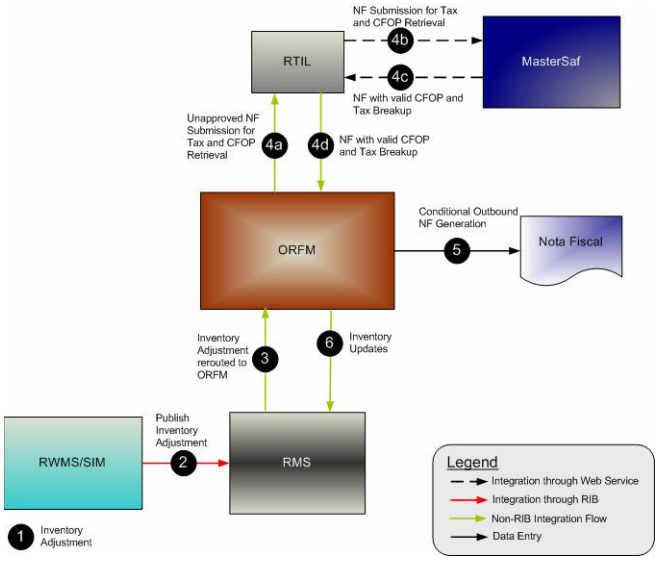
Inventory Adjustment

In Brazil, the inventory adjustment may also require NF generation, depending on the reason for which the inventory was adjusted. The same inventory adjustment reason codes defined in RMS are available in RWMS and SIM. In ORFM, the utilization codes are associated with the reason codes. It is not mandatory that each reason code is associated with some utilization code, but if the reason code is associated with the reason code, an outbound NF is generated.

Because the inventory adjustment is done in SIM or RWMS, an Inventory Adjustment message should be published by SIM or RWMS. This message requires no changes to support Brazil specific flows. The message is consumed by RMS, and the subscription API routes it to ORFM in the Brazil environment. If the reason code associated with the inventory adjustment requires NF generation, an outbound NF is generated. After the NF validation and approval, the inventory in RMS is updated. If the reason code associated with the inventory adjustment does not require NF generation, using the same information received in the Inventory Adjustment message, RMS inventory is updated.

Note: In Brazil, a positive inventory adjustment is not legal. Therefore it is assumed that the user will perform a negative inventory adjustment each time. The NF generated will then be an outbound NF.

Figure 4–9 Inventory Adjustment



Finishers/Repairing Flow (Two-Legged Transfer)

A retailer can send goods out to an external finisher or supplier for either finishing work such as printing, dying, or embroidery, or repair in the case of damaged goods. After the job is done, the finisher or supplier returns the goods to the same or a different location. While the goods are out of warehouse or store for repair or finishing, RMS needs to account for the inventory as the retailer's inventory.

This flow needs to be handled through two-legged transfers. The first leg of the transfer corresponds to the inventory being sent from the retailer's location (warehouse or store) to the finisher's location. The second leg corresponds to the receiving the finished goods (same or different SKU) or repaired goods back from the finisher at the retailer's location. Because the finisher's location is an outside location, the retailer is not responsible for the receiving or shipping at those locations.

Two-Legged Transfer Shipping from RWMS

When goods need to be sent to external finishers for finishing work, a two-legged transfer must be initiated in RMS, with transfer type as manual requisition. The first leg of the transfer (warehouse to finisher) flows to RWMS through RIB. RWMS persists it as a regular stock order. RWMS needs to do no special processing for such stock orders.

When goods need to be sent to external finishers or suppliers for repair, a two-legged transfer can be initiated in RMS, with the transfer type as manual requisition and the context type as repair. The first leg of the transfer (warehouse to finisher) flows to RWMS through RIB. RWMS stores it as a stock order with the type set to Repairing. All Repairing types of stock orders need special processing in RWMS. A regular wave should not select these stock orders for picking. These stock orders are picked manually, and only the inventory associated with appropriate trouble code can be picked up for repairing.

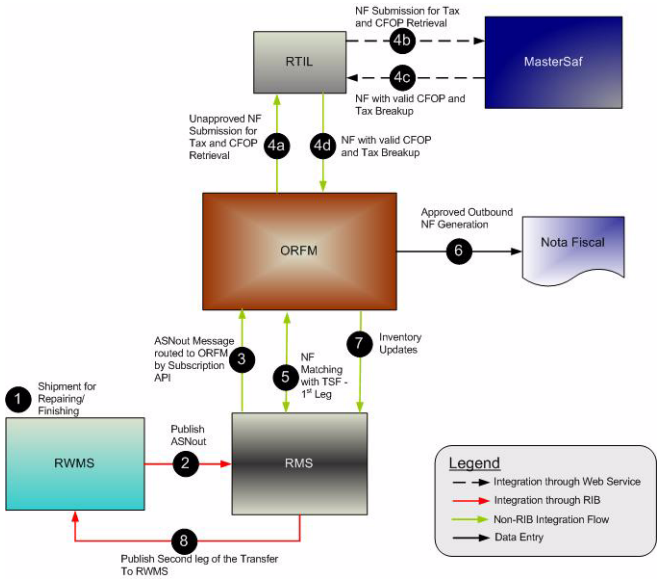
RWMS, as the physical owner of the inventory, has better visibility to the goods that need to be repaired. Therefore, a repairing stock order can be initiated in RWMS as well. As a stock order with the type of Repairing is created in RWMS, it flows to RMS through RIB. RMS, on consuming the message, creates a two-legged transfer with a context type of Repairing with the final location as the same origin warehouse.

As the required stock is picked and shipped at RWMS (for the external finisher's location), the ASNout message is published. ORFM consumes this message through the RMS subscription API and generates an outbound NF with Repairing as the requisition type. During NF validation, tax information is retrieved from Mastersaf, and NF matching is done with the requisition document. RMS inventory is updated, and then the NF is approved. On NF approval, the shipment is dispatched from the warehouse. RMS publishes the information of the second leg to RWMS through RIB subsequent to the inventory updates. RWMS does not maintain any link between the first and the second legs of the transfer. For RWMS, the two legs of the transfer should be two independent transactions. The second leg does not need to have context type as Repairing. If it is, RWMS ignores it if the recipient is the warehouse.

If the Auto Receive Stock indicator is enabled for the supplier in RMS, the on-hand inventory at the finisher's location is increased immediately as the stock is shipped from the retailer's location. If the indicator is disabled, on shipment of stock from retailer's location, in-transit inventory is increased at the external location. The on-hand inventory is updated only after RMS receives a receipt message from the finisher.

As the shipment leaves the warehouse, the first leg of the transfer is concluded from an ORFM perspective.

Figure 4–10 RWMS Two-Legged Transfer - First Leg (Shipping)



Two-Legged Transfer Receiving at RWMS

Receiving a two-legged transfer (second leg) is similar to transfer receiving or WO return receiving. The only difference is that the primary ASN is not already available in RWMS when the NF and schedule is entered in ORFM. The finisher ships the finished or repaired items (same or different SKUs) back to the warehouse along with the NF. As the shipment reaches the warehouse, a schedule is created and the inbound NF is entered into the system. The requisition number for the NF should be the main transfer number (two-legged). Each time, the requisition number on the NF should be the one in RMS. As the NF and schedule go through the tax validation and matching process, the schedule is submitted to RWMS for physical receiving. The message that flows from ORFM to RWMS through RIB should be equivalent to an ASNout message and should contain the schedule number and all the ASN details. The message should be detailed and should contain following fields:

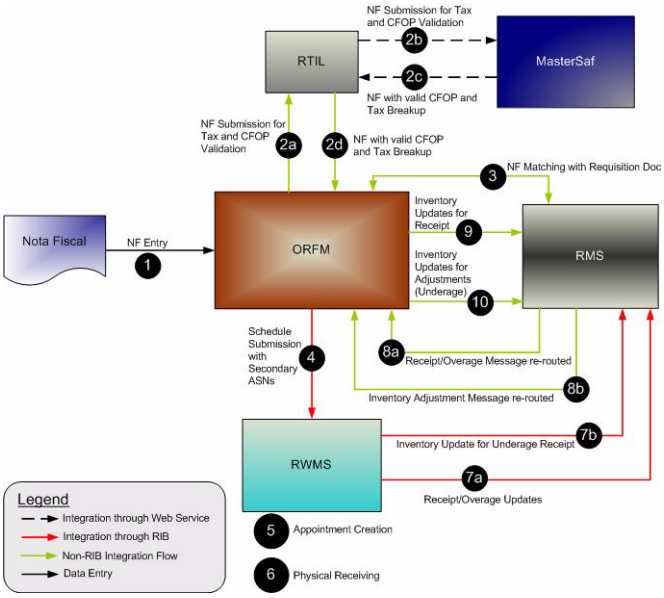
- ASN Number (ORFM should generate the ASN# and maintain the next up sequence number for it).
- Requisition Number (the secondary transfer number)
- Receiving Warehouse Number
- Distinct Item Ids per transfer leg two
- Consolidated Quantities per item-requisition
- UoM
- Schedule number
- Container Id (ORFM should generate the container number and should maintain the next up sequence)

As RWMS receives this message, an automatic appointment (per schedule) is created and the ASN is linked to the appointment. After scheduling the appointment, physical

receiving takes place. The user may find discrepancies between the NF and the physical receipt. In case of an overage, RWMS sends a separate overage message. Any over-receipt at RWMS is handled out of the system. In case of under-receipt, the receipt message coming from RWMS contains the same information as the ASN (items and quantities). Any inventory received short should flow to ORFM (through RIB and RMS subscription APIs) as a separate inventory adjustment message.

After ORFM consumes the receipt message, the inventory updates are done in RMS. The in-transit inventory at the destination location is reduced and on-hand inventory is increased. WAC is recalculated for the destination location. Subsequent to the inventory updates for receipts, the inventory should be updated for the inventory adjustment for any under-receipt.

Figure 4–11 RWMS Two-Legged Transfer - Second Leg (Receiving)



Two-Legged Transfer Shipping from SIM

When goods need to be sent from a store to an external finishers for finishing work, a two-legged transfer needs to be initiated in RMS, with the transfer type set as manual requisition and no context type. The first leg of the transfer (store to finisher) should flow to SIM through RIB. SIM should persist this as a return to warehouse (RTW) request.

When goods need to be sent out to external finishers or suppliers for repair, a two-legged transfer can be initiated in RMS, with the transfer type set to manual requisition and the context type to Repairing. The first leg of the transfer (warehouse to finisher) flows to SIM through RIB. SIM persists this as a return to warehouse request.

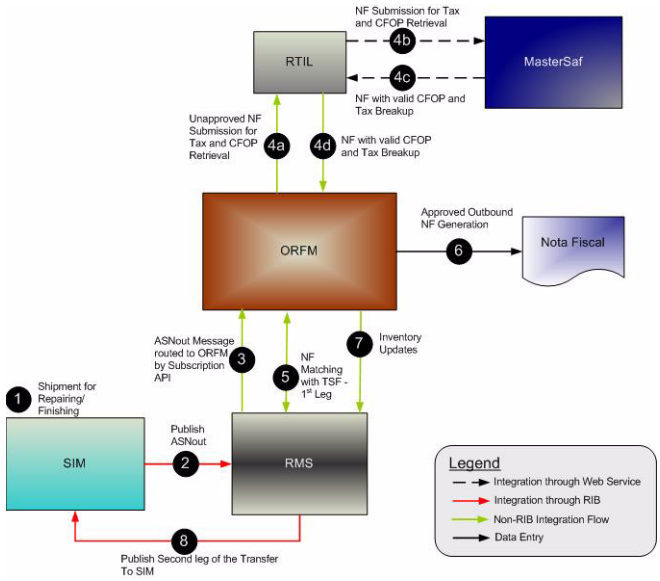
Because SIM is the physical owner of the inventory, it has better visibility to the goods that need to be repaired. Therefore, a repairing RTW can be initiated in SIM as well. As a RTW with the type set to Repairing is created in SIM, it should flow to RMS through RIB. RMS, on consuming the message, creates a two-legged transfer, with the context type set to the value from the transfer created in SIM, and the final location as the same origin store..

As the required stock is shipped from the store (for the external finisher's location), the ASNout message is published. RMS consumes this message and the subscription API directs it to ORFM. ORFM generates an outbound NF with Repairing as the requisition type. During NF validation, tax information is retrieved from Mastersaf. NF matching is done with the requisition document and on NF approval, the shipment is dispatched from the warehouse. RMS inventory is updated on NF approval. RMS publishes the information of the second leg to SIM through RIB after the inventory updates. SIM needs to maintain a link between the first and the second legs of the transfer because of serial-numbering requirements. RMS, while publishing the second leg to SIM, references the original transfer number. The second leg does not need to have the context type set to Repairing. SIM persists this transfer as a warehouse delivery.

If the Auto Receive Stock indicator is enabled for the supplier in RMS, the on-hand inventory at the finisher's location is increased immediately as the stock is shipped from the retailer's location. If the indicator is disabled, on shipment of stock from retailer's location, in-transit inventory is increased at the external location. The on-hand inventory is updated only after RMS receives a receipt message from the finisher.

As the shipment leaves the store, the first leg of the transfer is concluded from the ORFM perspective.

Figure 4–12 SIM Two-Legged Transfer - Shipping



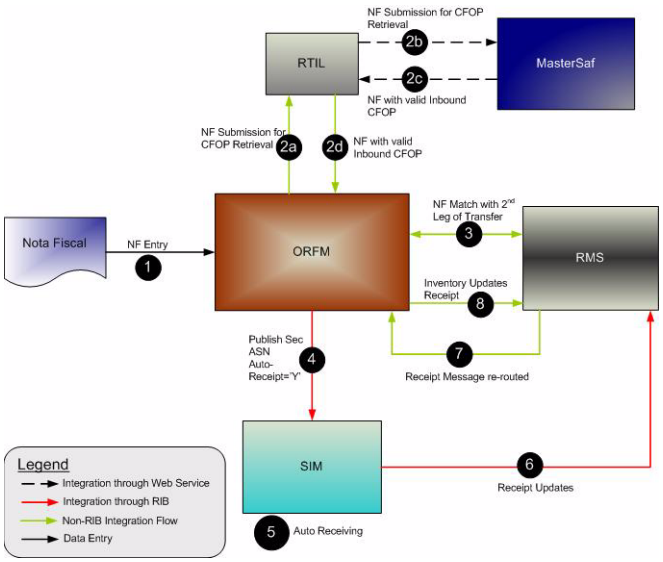
Two-Legged Transfer Receiving at SIM

Receiving a two-legged transfer (second leg) is similar to a transfer receiving at a store. The only difference is that the primary ASN is not already available in SIM when the NF and schedule is entered in ORFM. The finisher ships the finished or repaired items (same or different SKUs) back to the warehouse along with the NF. As the shipment reaches the store, a schedule is created and the inbound NF is entered into the system. The requisition number for the NF should be the main transfer number (two-legged). As the NF and schedule go through the tax validation and matching process, the ASN is published to SIM for physical receiving. The message that flows from ORFM to SIM through RIB should be the equivalent of an ASNout message. The auto-receiving indicator should be 'Y' for this ASN. The detailed message should contain the following information:

- ASN Number (ORFM should generate the ASN number and maintain the next up sequence number for it).
- Requisition Number (the secondary transfer number)
- Receiving store id
- Distinct Item IDs per transfer leg-two
- Consolidated quantities per item-requisition
- UOM
- Container ID (ORFM should generate the container number and should maintain the next-up sequence)
- Auto-receiving indicator (value should be 'Y')

As SIM receives this message, the ASN is automatically received and a receipt message is sent to ORFM. On receipt of the the receipt message from SIM, the inventory updates are done in RMS. The in-transit inventory at the destination location is reduced and on-hand inventory is increased. WAC is recalculated for the destination location.

Figure 4–13 SIM Two-Legged Transfer - Receive



5

ORFM Integration – Technical

This chapter is divided into the following two sections that address the ORFM methods of integration:

RIB-Based Integration

ORFM can integrate with other Orach Retail products, such as SIM and RWMS, through Oracle Retail Integration Bus (RIB). RIB utilizes a publish and subscribe (pub/sub) messaging paradigm with some guarantee of delivery for a message. In a pub/sub messaging system, an adapter publishes a message to the integration bus that is then forwarded to one or more subscribers. The publishing adapter does not know or care how many subscribers are waiting for the message, what types of adapters the subscribers are, what the subscribers current states are (running/down), or where the subscribers are located. Delivering the message to all subscribing adapters is the responsibility of the integration bus.

See the *Oracle Retail Integration Bus Operations Guide* and other RIB documentation for additional information.

Web Service-Based Integration

ORFM integrates with RTIL through webservice based interfaces. See "RTIL Architecture" on page 8-5 for additional information.

6

Publication and Subscription API Designs

This chapter includes publication designs that describe, on a technical level, how ORFM publishes messages to the Oracle Retail Integration Bus (RIB).

Secondary ASNOut Publication API

Business Overview

This message will be published from ORFM while submitting a schedule based on 'PO' to SIM for receiving or 'Transfer' to RWMS/SIM for receiving. RWMS/SIM subscribes to this message to do the receiving.

This RIB message is triggered by submitting a schedule in ORFM. The output message is in hierarchical structure, with ASN info in the Description section, distro, item, and carton info in the detail sections.

To facilitate the routing of data, the header level of routing info contains the 'to_phys_loc' with the value of location id and 'to_phys_loc_type' with the value 'W'/'S' and 'from_phys_loc' and 'from_phys_loc_type' information. Detail level routing info contains the 'source_app' with the value of 'ORFM'. This allows RIB to route the message from ORFM to RWMS/SIM.

Functionality Checklist

Table 6–1

Description	ORFM	RIB
ORFM must publish ASNOut information		
Create new publisher	X	X

Form Impact

None

Business Object Records

None

Package Impact

Business Object ID

The business object ID for ASNOut publisher is recv_no i.e. schedule no and po_ number i.e. the transaction number like PO/TSF number.

Package Name: FM_SECONDARY_ASNOUT

Spec File Name: fm_secondary_asnout_sql_s.pls

Body File Name: fm_secondary_asnout_sql_b.pls

Package Specification - Global Variables

L_FAMILY constant varchar2(25) := 'asnout'

Function Level Description -

ADDTOQ -The inbound Schedule for a transfer is published from ORFM as one ASNout message to SIM/RWMS and PO published to SIM.

This function adds a record in the FM_RIB_RECEIVING_MFQUEUE table with "asnoutmod" message_type for transfer, "asnoutcre" for PO/Two-legged transfer/Repairing and "ASNOut" family for each inbound schedule/ASN. The published flag should be set to 'U'.

GETNXT - The RIB calls GETNXT function to get ASNOut messages. It performs a cursor loop on the unpublished records with PUB_STATUS = 'U' and FAMILY='asnout' '' on the FM_RIB_RECEIVING_MFQUEUE table. For each record retrieved, GETNXT gets the following:-

- A lock on the queue table for the records which ORFM is going to publish. The lock is obtained by calling the function LOCK_THE_BLOCK.
- The information from the FM_RIB_RECEIVING_MFQUEUE table is passed to PROCESS_QUEUE_RECORD function. This function will build the Oracle Object message for RIB.
  - If PROCESS_QUEUE_RECORD does not run successfully, GETNXT raises an exception call to HANDLE_ERRORS function that updates the PUB_STATUS to 'H'.

PUB_RETRY - Same as GETNXT it performs a cursor loop on the records with PUB_ STATUS = 'H' and FAMILY = 'asnoutmod'. For each record retrieved, PUB_RETRY gets the following :-

- A lock on the queue table for the records which ORFM is going to publish. The lock is obtained by calling the function LOCK_THE_BLOCK.
- The information from the FM_RIB_RECEIVING_MFQUEUE table is passed to PROCESS_QUEUE_RECORD function. This function will build the Oracle Object message for RIB.
  - If PROCESS_QUEUE_RECORD does not run successfully, PUB_RETRY raises an exception call to HANDLE_ERRORS function that updates the PUB_ STATUS to 'H'.

PROCESS_QUEUE_RECORD (local) - This function is called from GETNXT and PUB_RETRY functions for those Inbound schedule records that are in PUB_STATUS = 'U' and 'H' respectively. This function calls API_LIBRARY.GET_RIB_SETTINGS to get the RIB settings for the ASNOut message family. It also calls BUILD_HEADER_OBJECT function to build the header and detail sections of secondary ASNOut message. Once the Oracle object is successfully formulated, this function deletes the current record from the queue (i.e. FM_RIB_RECEIVING_MFQUEUE table) by calling DELETE_QUEUE_REC function.

BUILD_HEADER_OBJECT (local) - This function will take necessary data from FM_RIB_STG_RECEIVING_HEADER and FM_RIB_STG_RECEIVING_DETAIL tables for the current schedule_nbr/ASN and prepare the Oracle object for the ASNOut message. The structure of ASNOut is same as base payloads with an additional attribute containing schedule_nbr in the "RIB_ASNOutDesc_REC" section for RWMS.

The same function will build ASNOut message structure with an additional attribute of auto_receipt flag set to 'Y' in the "RIB_ASNOutDesc_REC" section for SIM. This flag will otherwise be NULL in case of RWMS.

While publishing the secondary ASNOut to RWMS and SIM, ORFM will be sending three routing information through Oracle Object - RIB_ROUTINGINFO_REC.

1. Sending location
2. Receiving location
3. A hard-coded string depicting source application - 'ORFM'

This is required to indicate the source of ASNOut message is ORFM and hence, it is meant only for RWMS and SIM applications to consume it. However, RMS also subscribes to ASNOut message, but in this case as the source is ORFM - RMS subscriber package will discard the message.

Call the BUILD_DETAIL_OBJECTS to get the details of the current schedule record. The container_qty is a required field on the RIB object. So, ORFM sends it as 1 instead of NULL.

This function will also be modified to publish the following additional fields in the publishing message to RWMS/SIM :

1. Schedule number (will be NULL for SIM)
2. Auto_receive flag (will be NULL for RWMS and 'Y' for SIM)

BUILD_DETAIL_OBJECTS (local) - This function is responsible for building the detail section of ASNOut message. It builds as many detail Oracle Object as many items present in a transfer. It will fetch the detail records from FM_RIB_STG_RECEIVING_HEADER, FM_RIB_STG_RECEIVING_DETAIL for the given schedule number and will assign the above details into "RIB_ASNOutItem_REC", RIB_ASNOutCtn_REC" and "RIB_ASNOutDistro_REC" record groups.

This package will be modified to generate the ASN# and CTN# and will maintain the next up sequence number for it in case of PO to SIM and two-legged transfer/repairing publishing to RWMS/SIM.

LOCK_THE_BLOCK (local) - This function locks all queue records for the current business object. This is to ensure that GETNXT does not wait on any business processes that currently have the queue table locked and have not committed.

HANDLE_ERRORS (local) - HANDLE_ERRORS is called from GETNXT and PUB_RETRY when an exception is raised. If the error is a non-fatal error, GETNXT passes the sequence number of the driving FM_RIB_RECEIVING_MFQUEUE record back to the RIB in the ROUTING_INFO. It sends back a status of 'H'ospital to the RIB as well.

It then updates the status of the queue record to 'H'ospital, so that it will not get picked up again by the driving cursor in GETNXT.

DELETE_QUEUE_REC (local) - DELETE_QUEUE_REC is called from PROCESS_QUEUE_RECORD once a queue record is formed from FM_RIB_RECEIVING_MFQUEUE, FM_RIB_STG_RECEIVING_DETAIL and FM_RIB_STG_RECEIVING_HEADER tables. This is just to not pick the same record again

Trigger Impact

Trigger Name:
N/A

Trigger File Name:
N/A

Table:
N/A

Message XSD

Here is the filename that corresponds with the message type. Please consult the RIB documentation for this message type in order to get a detailed picture of the composition of the message.

Table 6–2

Message Types	Message Type Description	XML Schema Definition (XSD)
ASNOutCre	ASNOut Create message	ASNOutDesc.xsd
ASNOutMod	ASNOut Modify message	ASNOutDesc.xsd

Table Impact

Table 6–3

TABLE	SELECT	INSERT	UPDATE	DELETE
FM_RIB_RECEIVING_MFQUEUE	Y	Y	Y	Y
FM_RIB_STG_RECEIVING_HEADER	Y	Y	N	Y
FM_RIB_STG_RECEIVING_DETAIL	Y	Y	N	Y
FM_RECEIVING_HEADER	Y	N	N	N
FM_RECEIVING_DETAIL	Y	N	N	N

Design Assumptions

This message is applicable only if ORFM exists in case of Brazil countries. It is applicable for PO/Schedule publication to SIM and transfer/schedule publishing to RWMS/SIM.

POSchedule Publication API

Business Overview

This message is published from ORFM while submitting a schedule based on 'PO' to RWMS for receiving. RWMS subscribes to this message it order to create schedule based appointment and does the receiving.

This RIB message is triggered by submitting of PO based Schedule in ORFM. The output message is in hierarchical structure, with Schedule no in the Description section, PO information in the Header and Item details in the detail section.

To facilitate the routing of data, the header level of routing info contains the 'to_phys_loc' with the value of location id and 'to_phys_loc_type' with the value 'W'. Detail level routing info contains the 'source_app' with the value of 'ORFM'. This allows the RIB to route the message from ORFM to RWMS.

Functionality Checklist

Table 6–4

Description	ORFM	RIB
ORFM must publish POSchedule information		
Create new publisher	X	X

Form Impact

None

Business Object Records

None

Package Impact

Business Object ID

The business object id for POSchedule publisher is recv_no i.e. schedule no.

Package Name: FM_SCHED_SUBMIT

Spec File Name: fm_sched_submit_sql_s.pls

Body File Name: fm_sched_submit_sql_b.pls

Package Specification - Global Variables

FAMILY constant varchar2(25) := 'poschedule'

LP_cre_type varchar2(15) := 'poschedulecre'.

Function Level Description

ADDTOQ - This function adds the PO Schedule to the FM_RIB_RECEIVING_MFQUEUE table in 'U'npublished status. It stages the recv_no(schedule number) for publishing to the RIB.

GETNXT - This procedure is called from the RIB to get the next 'U'npublished schedule number from FM_RIB_RECEIVING_MFQUEUE table for publishing. Based on the seq_no on FM_RIB_RECEIVING_MFQUEUE, it calls the PROCESS_QUEUE_RECORD procedure.

GETNXT gets the following:

- A lock on the queue table for the records which ORFM is going to publish. The lock is obtained by calling the function LOCK_THE_BLOCK.
- The information from the FM_RIB_RECEIVING_MFQUEUE table is passed to PROCESS_QUEUE_RECORD function. This function will build the Oracle Object message for RIB.

  If PROCESS_QUEUE_RECORD does not run successfully, GETNXT raises an exception.
- If any exception is raised in GETNXT, including the exception raised by an unsuccessful call to PROCESS_QUEUE_RECORD, HANDLE_ERRORS is called.

PUB_RETRY - Same as GETNXT and it will process the record for PUB_STATUS = 'H' and MESSAGE_TYPE = 'poschedcre'. For each record retrieved, PUB_RETRY gets the following:-

- A lock on the queue table for the records which ORFM is going to publish. The lock is obtained by calling the function LOCK_THE_BLOCK.
- The information from the FM_RIB_RECEIVING_MFQUEUE table is passed to PROCESS_QUEUE_RECORD function. This function will build the Oracle Object message for RIB.
- If PROCESS_QUEUE_RECORD does not run successfully, PUB_RETRY raises an exception . Call to HANDLE_ERRORS function that updates the PUB_STATUS to 'H'.
- If any exception is raised in GETNXT, including the exception raised by an unsuccessful call to PROCESS_QUEUE_RECORD, HANDLE_ERRORS is called.

PROCESS_QUEUE_RECORD (local) - This function is called from GETNXT and PUB_RETRY functions for those Inbound schedule records that are in PUB_STATUS = 'U' and 'H' respectively. This function calls API_LIBRARY.GET_RIB_SETTINGS to get the RIB settings for its family ('posched'). It also calls BUILD_HEADER_OBJECT function to build header and detail sections of the PO Schedule Oracle objects. Once the Oracle object is successfully formulated, this function deletes the current record from the queue (i.e. FM_RIB_RECEIVING_MFQUEUE table) by calling DELETE_QUEUE_REC function.

BUILD_HEADER_OBJECT (local) - Take all necessary data from FM_RIB_STG_RECEIVING_HEADER and FM_RIB_STG_RECEIVING_DETAIL tables for the current schedule and put it into a "RIB_POScheduleDesc_REC" object. Two routing information (source Application i.e. ORFM and location i.e. Warehouse) has to be sent to RIB through RIB_ROUTINGINFO_REC for PO.

This function publishes the following fields in the header section of the POSchedule message to RWMS:

- schedule_nbr
- receiving_location_id

This function will give a call to the BUILD_DETAIL_OBJECTS function.

BUILD_DETAIL_OBJECTS (local) - The function is responsible for building detail level Oracle Objects. It fetches the detail records from FM_RIB_STG_RECEIVING_DETAIL for the given schedule number and formulates the message structures: "RIB_POSchedule_REC" and "RIB_POScheduleDtl_REC".

This function publishes the following fields to the above Oracle objects:

- requisition_nbr
- requistion_type
- tem_id
- consolidate_qty

LOCK_THE_BLOCK (local) - This function locks all queue records for the current schedule_nbr. This is to ensure that GETNXT does not wait on any business processes that currently have the queue table locked and have not committed.

HANDLE_ERRORS (local) - HANDLE_ERRORS is called from GETNXT and PUB_RETRY when an exception is raised. If the error is a non-fatal error, GETNXT passes the sequence number of the driving FM_RIB_RECEIVING_MFQUEUE record back to the RIB in the ROUTING_INFO. It sends back a status of 'H'ospital to the RIB as well. It then updates the status of the queue record to 'H'ospital, so that it will not get picked up again by the driving cursor in GETNXT.

If the error is a fatal error, a status of 'E'rror is returned to the RIB.

DELETE_QUEUE_REC (local) - DELETE_QUEUE_REC is called from PROCESS_QUEUE_RECORD once a queue record is formed from FM_RIB_RECEIVING_MFQUEUE, FM_RIB_STG_RECEIVING_DETAIL, FM_RIB_STG_RECEIVING_HEADER tables. This is just to not pick the same record again

Trigger Impact

Trigger Name:

N/A

Trigger File Name:

N/A

Table:

N/A

Message XSD

Here is the filename that correspond with the message type. Please consult the RIB documentation for this message type in order to get a detailed picture of the composition of the message.

Table 6–5

Message Types	Message Type Description	XML Schema Definition (XSD)
POScheduleCre	POSchedule Create message	POScheduleDesc.xsd

Table Impact

Table 6–6

TABLE	SELECT	INSERT	UPDATE	DELETE
FM_RIB_RECEIVING_MFQUEUE	Y	Y	Y	Y
FM_RIB_STG_RECEIVING_ HEADER	Y	Y	N	Y
FM_RIB_STG_RECEIVING_ DETAIL	Y	Y	N	Y
FM_RECEIVING_HEADER	Y	N	N	N
FM_RECEIVING_DETAIL	Y	N	N	N

Design Assumptions

This message is applicable only if ORFM exists for Brazil localization. It is applicable for PO/Schedule publication to RWMS only.

7

Archiving and Purging Strategy

In a production environment, the number of transactions increases over a period of time. To keep performance acceptable, data must be purged periodically from the active tables of the application. ORFM purges the data from the active tables and stores the data in history tables.

Archiving and Purging

To enter a Nota Fiscal (for a PO, TO, or RTV) a schedule must be created in ORFM. After the schedule is in Financial Posted status and passes a certain number of days, all related data with that schedule are purged from the active tables. In addition, all the NFs that are inactive for a schedule are purged. An inactive schedule is also purged from the active table and the data is stored in the history table.

8

Integration with Mastersaf - Retail Tax Integration Layer

Brazil has complex tax and fiscal systems. To have a flexible format to fit any kind of tax scenario, and also to make the tax rules setup part of a specialist solution, Oracle Retail has integrated its applications with third-party tax engines provided by Oracle partners. The ORFM module is integrated with the Mastersaf Tax Rules solution through the Retail Tax Rules integration layer.

The Retail Tax Rules Integration layer (RTIL) acts as a connector that exposes a Retail Tax Canonical Data Model API for calculating tax, which is largely independent of any vendor-specific tax data model. RTIL hosts adapters that are responsible for converting the canonical model into the native language of a tax service provider and communicating with a third-party tax solution service. RTIL supports integration of RMS and ORFM with Mastersaf. The ORFM module is integrated with the Mastersaf Tax Rules solution to obtain information about all Brazilian tax legislation, with a high level of exception treatments.

For all flows in Oracle Retail that need to have tax calculation, the integration layer is used to have all taxes applied from the Mastersaf Tax Rules (MTR), considering the input parameters.

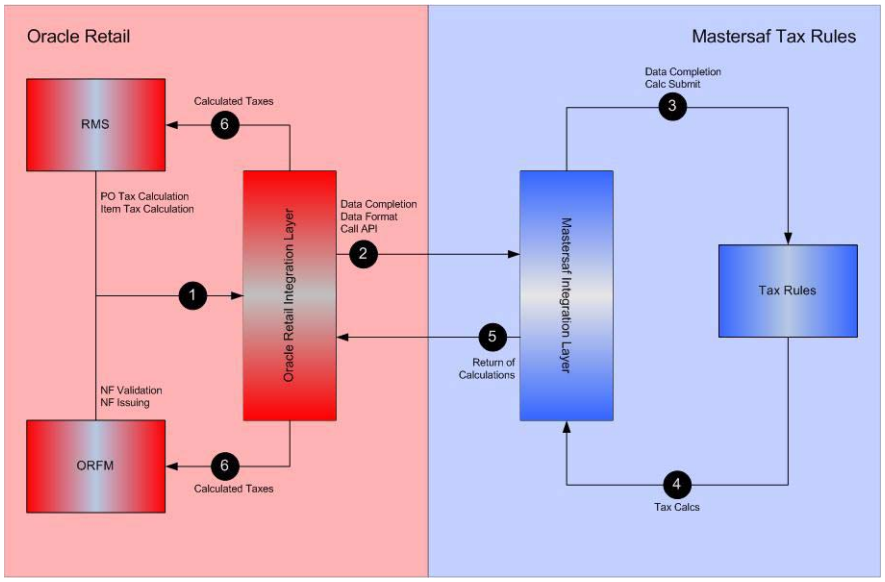
The following processes in ORFM/RMS use the tax calculation integration:

- Inbound NF validation
- Outbound NF issuance
- PO tax breakdown
- Item creation

For each of these processes, the needed input information is sent to MTR thru RTIL. RTIL is responsible for transforming the request information from the canonical tax data model format to Mastersaf specific format, and vice versa for the response information.

Integration Overview

Figure 8–1 Oracle Retail/Mastersaf Integration



RTIL is responsible for the Mastersaf API call with the necessary formatting. The Mastersaf API exposed has a native Nota Fiscal format, and formatting must be sent in that way for tax calculation in MTR.

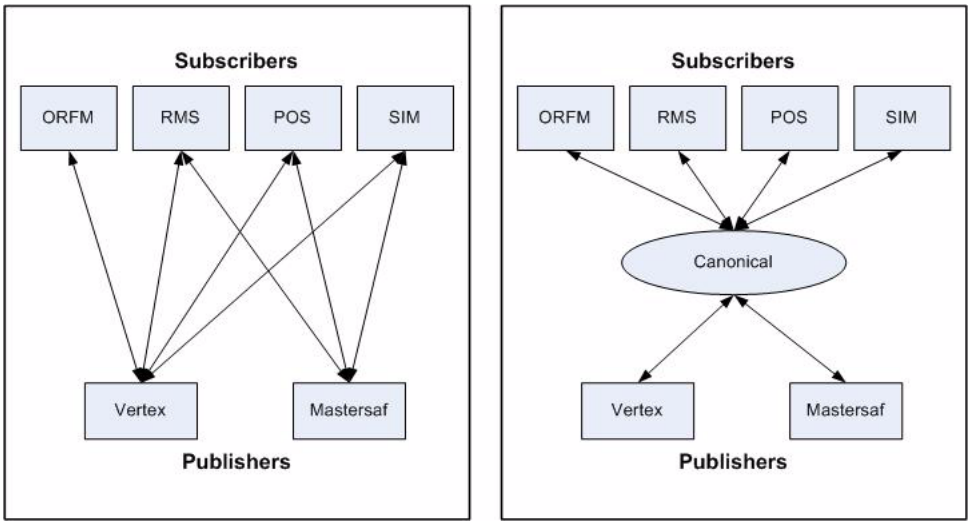
In MTR, an integration layer is used for data completion. Some of the attributes in theAPI must be determined by Tax Rules (such as CFOP). Only after this step is complete the tax classification is triggered internally in MTR. Once the taxes are classified, the API with the returned values is sent back to the integration layer in Oracle Retail.

The layout of MTR solution is based on the NF layout and has the following structure:

- Entities (foundation data and fiscal attributes of each entity of a NF, such as Issuer,
- Addressee, and Transporter)
- NF header information
- NF item information
- Tax classification ("enquadramento")

Retail Tax Data Model - Need for Canonical Data Model

Figure 8–2 Canonical Data Model



The canonical model is a paradigm for achieving semantic interoperability. The canonical model aims to decouple systems at the level of message formats. Systems do not have to rely on the data formats of other systems. The canonical model allows applications to be loosely coupled.

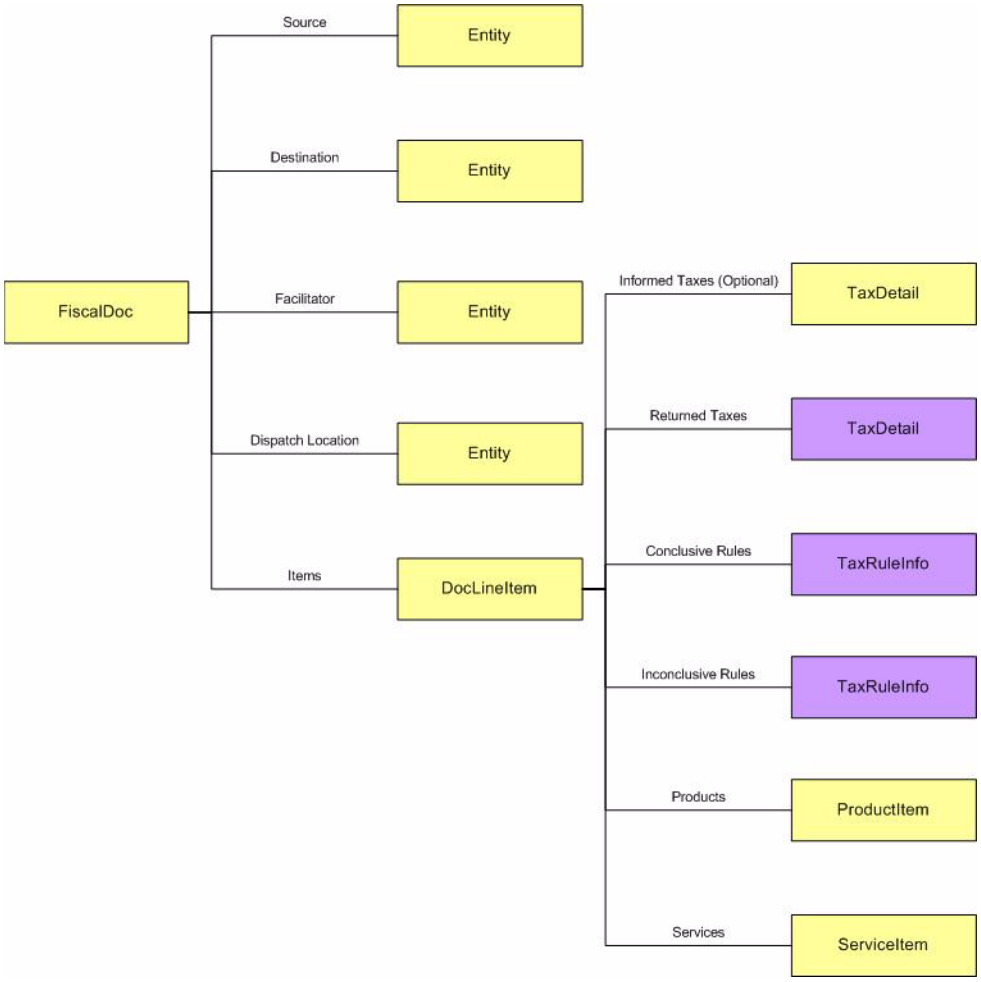
Benefits of Canonical Data Model

- Using canonical message types decouples systems at the level of message formats. Systems do not have to make assumptions or rely on the data formats of other system. This is an important aspect to achieve loose coupling.
- Defining canonical message formats creates the opportunity to supply the company with an unambiguous catalog of available messages about business events, representing valuable business assets. The business events in this catalog are independent from the sources that generate these messages.
- This pattern has a benefit in that at the endpoints, only one transformation service per message type has to be configured. A subscriber needs to subscribe to only one message type, regardless of whether there are multiple sources or not.
- If transformations take place directly between local formats (skipping the intermediate canonical format), transformation services have to be created for every source-target combination. This leads to higher loads of management and maintenance efforts. Consumers would have to subscribe separately to every source of a particular message type and have knowledge of the existence of these distinct sources.
- Without an intermediate canonical format, a format change at the publisher side must be followed by changing all the transformations to the subscribers. Using an intermediate canonical format makes the transformations to the subscribers independent of changes at the publisher side.

- Without canonical formats for semantics representation, semantics would be represented in multiple equivalent formats. This is an obstacle to supplying the company with an unambiguous catalog of business events independent from their sources. The lack of canonical formats consequently can cause system designs and resulting systems to be more complex and harder to change

Retail Tax Data Model

Figure 8–3 MTR API Structure



Object Structure Overview

- FiscalDoc: This object represents a fiscal document and eventually maps to the Nota Fiscal of Mastersaf in the integration with MTR. It has the information related to the NF header and it has included the other objects that detail each part of the fiscal document as the issuer, addressee, and items.
- Entity: This object has the fiscal and master information related to the entities in a fiscal document. This object will be used to detail the issuer/supplier, addressee/location and facilitator/transporter information.
- DocLineItem: This object represents the detail of a line item in a fiscal document. It has included the taxes at item level.

- TaxDetail: This object represents the details of an individual tax. It can be attached at the Fiscal Document level or at a DocLineItem Level. Each tax applied to an item will have a different object of this type.
- TaxRuleInfo: This object has the rules and laws applied to each tax. It has the regulation and the log of the tax classification. It can be used in two different ways: First, to demonstrate the rules applied to the item (conclusive rules), and second, to demonstrate the rules not applied to the item and classified as inconclusive rules. This last one can be used as an information log to the user to show potential tax setup problems.
- ProductItem/ServiceITem: Code and description of items which can be products or services. These objects will have the fiscal attribute details for the items depending on the item type (product or service).

Retail Tax Data Model to Mastersaf Data Model Object Mapping

The following table illustrates the mapping of the Retail Tax Data Model to Mastersaf at an object level.

Table 8–1

Retail Tax Object	Mastersaf Object
FiscalDoc	DocFiscal
Entity	Pessoa
DocLineItem	ItemDocFiscal
TaxDetail	Enquadramento
TaxRuleInfo	EnquadramentoItem
ProductItem	Produto
ServiceItem	Servico

RTIL Architecture

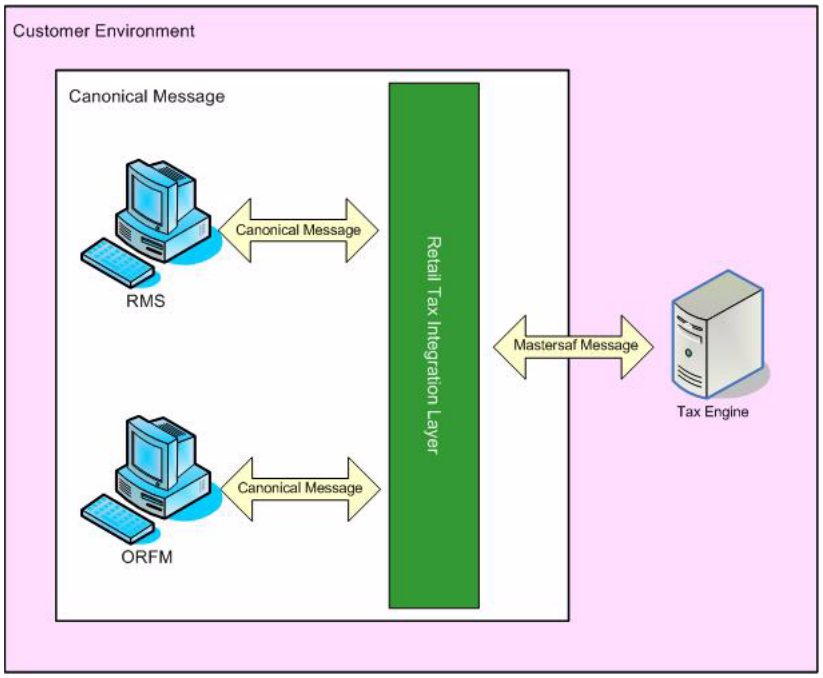
The canonical data model for tax calculation is exposed as a Web service operating in a synchronous request/response pattern. A Web service is chosen because it is loosely coupled and it allows different applications to talk to each other and share data and services.

The Retail Tax Integration layer is implemented as a Java EE application hosting the canonical Web services and associated tax service provider adapters. This layer forms the conduit between the Oracle Retail applications and the tax service provider. The Retail Tax Integration layer is responsible for assembling and disassembling the vendor-specific data model to the canonical tax data model, based on the configuration. The Retail Tax Integration layer hosts vendor-specific connectors that can communicate to the external third-party services with various protocols like EJB, JMS, SOAP, and HTTP, based on the configuration. The subscribing application should have a vendor-neutral canonical tax Web service callout utility for making Web service calls. The subscribing application is not aware of the tax service provider. RTIL acts as a bridge between the subscribing application and third-party tax service provider.

Note: Vertex, Synchro Adapter/Connecter Module is not in scope for 13.2 release.

High Level Integration View

Figure 8–4 RTIL High Level Integration

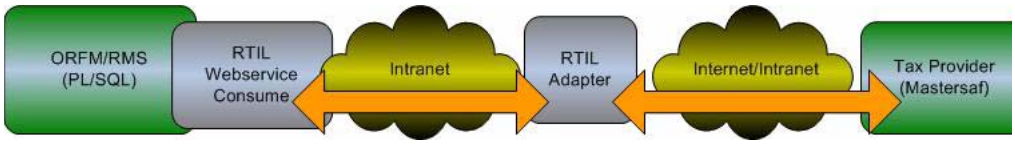


The integration layer in Oracle Retail does the following.

- Exposes a canonical (application-neutral) data format for RMS/ORFM to pass their transaction details that require taxes to be computed
- Performs the data transformation from canonical to Mastersaf specific format
- Invokes the taxation API of Mastersaf
- Transforms the response containing taxes from Mastersaf specific format back to the canonical format to be consumed by RMS/ORFM
- RTIL is used as the unique tax calculation entry point.

RTIL Integration Architecture

Figure 8–5 RTIL Integration Architecture

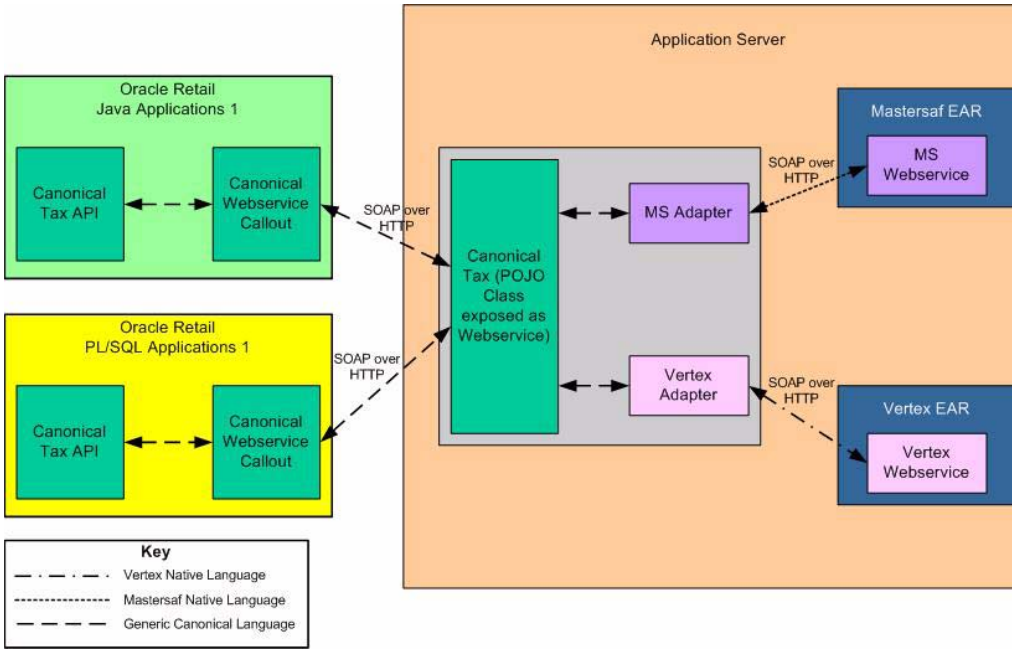


The above diagram illustrates the integration architecture of RTIL.

- RTIL is deployed as a J2EE application.
- RTIL exposes a Web-service-based interface for invocation from PL/SQL apps.
- It interacts with tax solution providers through application adapters
- Adapters perform the necessary data transformations between Oracle Retail applications and the tax vendor solutions.

Components Deployment View

Figure 8–6 Components Deployment View



Note: Integration with Vertex is not in scope for this release and is used for indicative purpose only to demonstrate the integration of a new tax provider. In addition, for the current release the integration is restricted to ORFM/RMS which are Forms based applications. The java applications are also depicted for informational purposes only.

The preceding diagram illustrates the deployment view of the components. The architecture is designed to be extensible, to allow for integration of additional tax service providers through the appropriate connectors/adapters. The features of this architecture are:-

- Retail applications interact only with canonical tax services in an application-neutral format (canonical model) using RTIL-supplied client components.
- RTIL insulates the Oracle Retail applications from tax service providers. RTIL will be deployed as an enterprise application instance (EAR file) in Oracle WebLogic Application Server.
- RTIL will host the vendor adapters/connectors.
- The adapters/connectors interact with the tax service providers in native format.
- The adapters are responsible for converting the canonical model to the application-specific data model and vice versa.
- Tax service providers can be deployed on the same or a different application server, based on customer needs.

Client Components

Canonical Tax API: This is a TaxAPI in canonical format which is exposed by RTIL to Oracle Retail application. Any data enrichment, if required, should be done prior to the invocation of this API call. This API will be in the language of the Oracle Retail application implementation.

For example, in the case of ORFM and RMS, the API is exposed as a PL/SQL package with functions for tax calculation. These APIs are generated in a standardized fashion, leveraging the Retail Service Enabler (RSE) utility.

Canonical Web service callout: This provides the run-time support for invoking a canonical API which is exposed as a Web service. The run-time artifacts are generated using the RSE utility. The current release is made up of PL/SQL Web service consumer artifacts generated by RSE.

RTIL Configuration

RTIL broadly supports the following configuration features:

- Logging/auditing
- Tax service provider URL configuration
- User-friendly exception messages

Logging/Audit

Logging is implemented in a declarative manner using Spring AOP (Aspect Oriented Programming). AOP is a paradigm that addresses separation of concerns, thereby ensuring that logging and business logic are kept separate in the code, avoiding code cluttering. Logging is introduced during run time in a noninvasive manner leveraging spring AOP features. The logging utilities are implemented using the industry-standard Apache commons logging API.

The following are the classes that act as the logging aspect to the application and apply to different methods of the application as configured.

- LoggerAspect
- ObjectLoggerAspect
- TimerLoggerAspect

LoggerAspect

This class logs entry and exit statements for the methods in core RTIL classes. This also logs if there is some exception during the execution of these methods.

ObjectLoggerAspect

This class logs the entire content of request and response objects used in the transactions. It also logs the request object content and response object contents both in the canonical and native Mastersaf format. This is useful for troubleshooting the request and responses.

TimerLoggerAspect:

This logs the time taken in miliseconds for execution of the methods present in assembler, webserviceGateway, and adapter classes. This aspect can be used to obtain a timing profile of the method executions.

These aspects are declared via annotations and registered in applicationContext.xml.

The following is a snapshot of the applicationContext.xml configuration file which shows the configuration for registering Aspects.

```
<aop:aspectj-autoproxy />
<bean id="timingLoggerAspect"
class="com.oracle.retail.tax.aspects.TimingLoggerAspect" />
<bean id="loggerAspect" class="com.oracle.retail.tax.aspects.LoggerAspect" />
<bean id="objectLoggerAspect"
class="com.oracle.retail.tax.aspects.ObjectLoggerAspect" />
```

Note: Removal of these entries in this configuration file is equivalent to remove the code which logs these details. Thus, the enablement/disablement of logging is controlled in a declarative manner using Spring AOP.

A second level of control is available in the log4j.properties file. This file controls the following logging features whose configuration can be changed to suit customer needs:

- Logging at different granularities (INFO, ERROR, DEBUG, FATAL)
- File size limit for each log file (log4j.appender.LOGFILE.MaxFileSize)
- Number of log files to be retained during logfile rollover (log4j.appender.LOGFILE.MaxBackupIndex)

The following is a snapshot of the log4j.properties configuration file supplied with default options during deployment time.

```
# Root Level Logger
log4j.rootLogger=INFO,STDOUT

log4j.appender.STDOUT=org.apache.log4j.ConsoleAppender
```

```
log4j.appender.STDOUT.layout=org.apache.log4j.PatternLayout
log4j.appender.STDOUT.layout.ConversionPattern=%d %p [%c] - %m%n

log4j.appender.LOGFILE=org.apache.log4j.RollingFileAppender
log4j.appender.LOGFILE.MaxFileSize=5MB
log4j.appender.LOGFILE.File=./log/rtil.log
# Keep ten backup files.
log4j.appender.LOGFILE.MaxBackupIndex=30
# Pattern to output: date priority [category] - message
log4j.appender.LOGFILE.layout=org.apache.log4j.PatternLayout
log4j.appender.LOGFILE.layout.ConversionPattern=%d %p [%c] - %m%n

#Class Level logger
log4j.category.org.dozer.MappingProcessor=INFO,STDOUT,appender
log4j.category.com.oracle.retail.tax.aspects.TimingLoggerAspect=DEBUG,LOGFILE
log4j.category.com.oracle.retail.tax.aspects.ObjectLoggerAspect=DEBUG,LOGFILE
log4j.category.com.oracle.retail.tax.aspects.LoggerAspect=DEBUG,LOGFILE

log4j.rootCategory= ALL,LOGFILE,STDOUT
```

Tax Service Provider Configuration

The endpoint URL configuration for Mastersaf is configured in tax-service-provider.properties and is found in rtil-config.jar of the ear file. The deployment team should be responsible for changing the contextpath and porttype based on the deployment details of the Mastersaf tax solution. Typically, the servername and port parameters are modified to reflect to the different deployment environments.

The following properties represent a sample tax-service-provider.properties file that capture the URL address of the deployed Mastersaf solution.

Mastersaf Endpoint URL configuration :

- ms.br.ws.serverName=mspdv310.us.oracle.com
- ms.br.ws.port=18001
- ms.br.ws.contextpath=taxrulesruntime
- ms.br.ws.porttype=TaxRulesAPI

These parameters are declared in tax-service-provider.properties. This tells about the address of mastersaf webservice.

User Friendly Exception Messages

RTIL provides some flexibility in configuring user-friendly error messages based on the exception conditions encountered. The ExceptionMessage.properties file contains the list of system-generated exception messages in RTIL, which can be further mapped to the user messages required by the customer. The file is a typical properties file containing key value pairs. This file is read by RTIL infrastructure to substitute the system generated message pattern (Key) with the supplied user message (value), and the user message is propogated to the calling infrastructure (ORFM/RMS).

The following is a snapshot of the ExceptionMessages.properties file.

```
java.net.UnknownHostException=Mastersaf is not configured properly in RTIL. Please
check with RTIL admin
java.net.ConnectException=Mastersaf is not configured properly in RTIL. Please
check with RTIL admin
Dozer\ Mapping\ Exception=Dozer Bean Mapping Exception
Failed\ to\ access\ the\ WSDL=Unable to connect to Mastersaf. Please check with
```

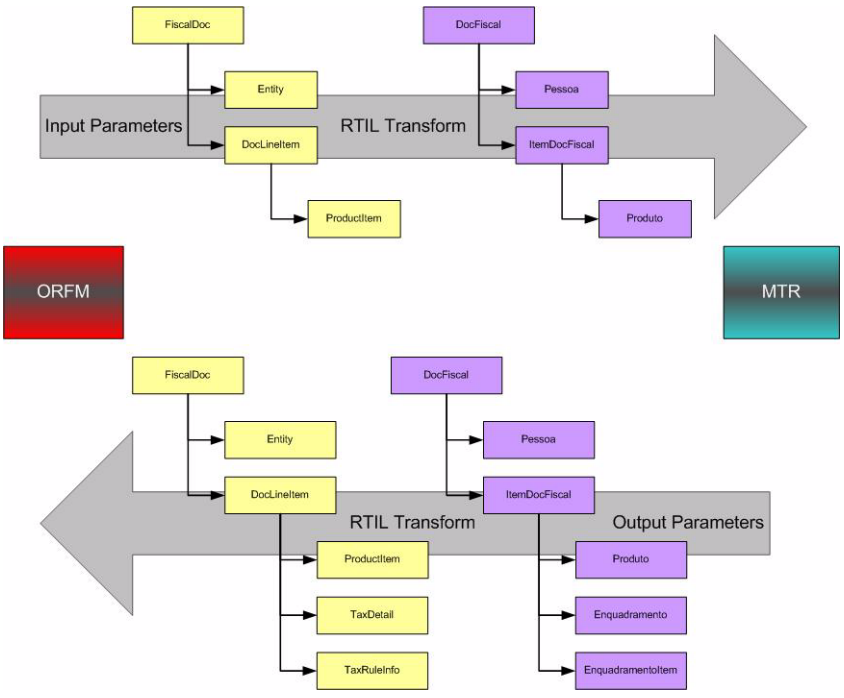
```
server Admin
Failed\ to\ read\ wsdl\ file=Unable to connect to Mastersaf. Please check with
server Admin
Exception\ while\ configuring\ TaxRulesAdapter=Unable to connect to Mastersaf.
Please check with server Admin
SchemaValidationErrors\ [2]\ for\ OUTBOUND=Bean validation Failed
TaxRulesException\ while\ connecting\ to\ ProxyService=Unable to connect to
Mastersaf. Please check with server Admin
TaxServiceProvider\ mapping\ configured\ for\ Country\ and\ Transaction\
TypeBrasil\:NONE =Country not configured properly
```

Note: The configuration option provided is only for supplying/altering the user defined messages. This file does not support the addition/modification of system defined messages.

Input Parameters Data Mapping and Expected Output Results

ORFM calls the Tax Rules API for the expected scenario and considering the structure of the API, which is similar to a fiscal document. All the input parameters are formatted in that structure, and the returned values are also sent in the same format. The input parameters do not contain any tax information. Only the fiscal attributes and values are sent. In the output parameters, the tax calculation results are sent back based on the original values in the same fiscal document format.

Figure 8–7 Input x Output Parameters



Tax Calculation Scenarios

For all the scenarios, the input parameters are based on the canonical API objects. Only the tax objects are not used, and the detail data-mapping for the input parameters can change depending on each scenario.

Scenario 1 - Inbound NF Validation

The inbound NF validation happens for NF receiving when ORFM calls the validate function. The necessity to validate a NF happens when an entire NF is input in ORFM, and the system must validate the informed taxes against the calculated. Informed taxes in ORFM come at the header level for manual NFs and also at the detail level for automatic NF (EDI or NFE).

Scenario 2 - Outbound NF Issuing

The NF issuing process can happen in ORFM in two different ways. The NF can be generated via EDI tables, as the transfers, RTVs and other outbound movements that are generated from RWMS and SIM, or the NF can be generated manually via ORFM, as the automatic return NF, rural producer NF and RMA. For both cases, the taxes are not informed and the tax calculation needs to come from MTR.

For the EDI NF generation, the batch that calculates the taxes in RMS through a call to the ORFM tax engine will be changed to call the new API with MTR. The NF must be already created in the fiscal doc tables with the correspondent number, because the NF number, NF ID and NF item ID are necessary to call the tax API. The process that calls the tax calculation is the same Validation process of ORFM screens.

Scenario 3 - PO Tax Calculation

The PO tax breakdown is the calculation of all applicable taxes of a PO. The PO information is similar to a fiscal document, and it is similar to the NF issuing process. The PO information utilizes the API format to have the tax calculated.

Scenario 4 - Item Creation

During the item creation process, the sales taxes applicable to each locationwhere the item is linked are calculated. The item creation process also makes a call to the tax API to get the sales tax rates. The same concept is applied to the default purchase taxes used for margin calculation.

Scenarios such as primary supplier change and fiscal reclassification call the same tax integration defined for original item creation process. In the Oracle Retail integration layer, the PO is formatted into the API layout. There are multiple API calls depending on the delivery location set on the PO. For each location, there is one call because the tax calculation API has the NF layout, and only one origin and destination is allowed per NF.

Scenario 5 - Freight NF Calculation

Freight NF has special characteristics in terms of tax calculation. It does not have any items, so the tax mapping considers default data.

Nota Fiscal Eletrônica (NFe)

Nota Fiscal Eletrônica (NFe) or Electronic Fiscal Note is a Brazilian government project with the objective of implementing a national model of electronic fiscal document to substitute the current system of issuing the fiscal documents in paper. The virtual document has juridical validity guaranteed by the digital signature of the issuer. It simplifies the fiscal obligations of the taxpayers and allows the follow-up of the commercial operations by the tax authority.

The NFe issuer generates an electronic file with all NF information in a more detailed level than the regular NF. This file must be digitally signed to guarantee the integrity of the data and the authorship of the issuer.

The electronic file that corresponds to the NFe is transmitted through the internet to the SEFAZ (Secretaria da Fazenda - Brazilian Tax Authority) of the origin state of the issuer. The SEFAZ provides a pre-validation of the file and returns a receiving protocol (Authorization for Use), that is necessary to the traffic of the goods.

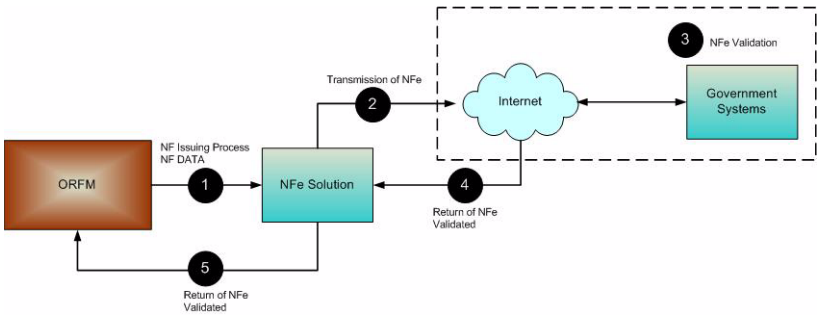
To follow the goods, a graphic representation of the NFe is printed. The DANFE (Documento Auxiliar da Nota Fiscal Eletrônica-Auxiliary Document of the Electronic Invoice) is printed in a common paper, one copy that highlights the access key for consultation of the NFe in the internet and a bi-dimensional bar code which facilitates the capture and confirmation of information of the NFe by the fiscal units.

The DANFE is not a Nota Fiscal, and does not replace the NF. It is just an auxiliary document for consultation of the NFe. It has the access code of the NFe which allows its owner to confirm the real existence of the NFe in RFB environment (Receita Federal Brasileira-Brazilian Federal Tax Authority) or the SEFAZ web site.

Overview

The overall solution landscape considers that ORFM works with a third party solution for NFe generation and transmission to the government.

Figure 8–8 NFe Issuing Solution Approach



NFe Options

ORFM allows the generation of the NFe by location and utilization code. A system option is used to identify the location that will issue the NFe and for which transaction, such as transfers, intercompany transfers, and RTVs.

Considering that the transactions where the NFe issuing is applicable, like transfers, are initiated in RWMS. Hence the utilization code associated to each location must be the same utilization code set in system variables as default utilization for outbound transactions, such as transfers, intercompany transfers, and RTVs. The default

utilization code is used by RWMS to generate the NFS. This behavior is not controlled in the system and must be defined by the user.

One location may use NFe for transfers but not for RTVs. While NF creation, ORFM considers this parameterization in order to enter in the NFe flow.

The document type in that case is auto-filled with the 55. That is the defined document type for NFe. That can also be set up as a system variable, so no hard codes are placed.

NFe Publishing

The FM_STG_NFE staging table for NFe contains the fiscal doc id and its status for the Java Adapter to fetch those records, which are in "NFe Pending" or "NFe Corrected" or "NFe Canceled" state and submit them to Mastersaf. The EVENT_ID field contains the sequence in which the NFe messages are published to Mastersaf. The fiscal_doc_no, series_no, cnpj and justification fields are used to allow Mastersaf's NFe product to interface the fiscal document information to SEFAZ without scanning through the object source again.

There are eight status codes to capture NFe flow between ORFM, Mastersaf, and SEFAZ.

- NFe_P - Fresh NFe document waiting to be picked by Mastersaf.
- NFe_X - Corrected NFe document waiting to be picked by Mastersaf.
- NFe_C - Canceled NFe document waiting to be picked by Mastersaf.
- C_A - Mastersaf updates the staging table with this status when NFe is successfully canceled.
- N_A - Mastersaf updates the staging table with this status when NFe is successfully nullified.
- NFe_I - Intermediate flag that is updated by Mastersaf while the document has been sent to SEFAZ, just in case of emission. A response to a cancel call is provided by SEFAZ immediately, whereas an emission situation does not provide an immediate call.
- A - Approved NFe from SEFAZ.
- E - Erroneous NFe from SEFAZ due to data/transmission errors (in any situation; emission, cancel or nullify).

NFe Subscription

On the consumption side, ORFM writes the API and grants access to Mastersaf to load acknowledgments (approval or erroneous details) from SEFAZ into the respective ORFM tables.

When the NFe gets approved successfully or processed with errors, Mastersaf makes a call to the ORFM packaged procedure FM_MS_NFE_SQL.CONSUME to update the NFe details into the respective ORFM tables. If there are any errors then it inserts the NFe transaction history table with error_id and error_description.

The consume procedure contains the following parameters:

Table 8–2

Variable Name	Input/Output Designation	Data Type/Length
O_status_code	IN OUT	NUMBER
O_error_message	IN OUT	VARCHAR2
I_message	IN	OBJ_MS_RFM_NFE_REC

I_message contains the following fields:

Table 8–3

OBJ_MS_RFM_NFE_REC	
Name	**Type**
fiscal_doc_id	number10
nfe_access_key	varchar244
nfe_protocol	number15
nfe_danfe_url	varchar21000
status	varchar26
errorDtl_tbl	OBJ_MS_RFM_ErrorDtl_TBL

Table 8–4

OBJ_MS_RFM_ErrorDtl_REC	
Name	**Type**
message_id	number4
message_desc	varchar21000
transaction_date	timestamp

OBJ_MS_RFM_ErrorDtl_REC is used to describe the errors (if any) that occurred in the process of NFe submission to SEFAZ.

When the NFe is approved by SEFAZ without any errors, Mastersaf makes a call to this CONSUME procedure with status as 'A', along with the corresponding values in other fields like NFE_ACCESS_KEY, NFE_PROTOCOL, and NFE_DANFE_URL with appropriate data. In this case, the 'errorDtl_tbl' will be NULL since there are no errors associated with it. The ORFM table is updated based on this input data.

If any errors occurred in NFe processing, the status will be 'E'. Now the 'errorDtl_tbl' will contain the error details. Here NFE_ACCESS_KEY, NFE_PROTOCOL, NFE_ DANFE_URL fields will be NULL.

When the fresh document (NFe_P) or corrected document (NFe_X) or canceled document (NFe_C) is picked from the staging table FM_ STG_NFE by Mastersaf's Java integrator monitor for polling, Mastersaf makes a call to this CONSUME procedure with status as 'NFe_I'. All the remaining fields will be NULL:-

- nfe_access_key
- nfe_protocol

- nfe_danfe_url
- errorDtl_tbl (PL/SQL table type)

If the CONSUME procedure called by Mastersaf is successful, then O_status_code will be 'S' (Success). If it is unsuccessul, it will be 'E' (Error) with the error referenced in the O_error_message.

There could be some network transmission errors during NFe flow between ORFM, Mastersaf, and SEFAZ. Such error codes are predefined by SEFAZ and do not require the user to correct anything. Mastersaf has provided two codes for network disruptions that can be resent without manual intervention, 286 and 296 . Logic in the FM_MS_NFE_SQL.CONSUMEsubscription API automatically resends such rejected NFes without manual intervention.

Sistema Público de Escrituração Digital (SPED)

Sistema Público de Escrituração Digital (SPED) or Public System of Digital Bookkeeping is the result of several efforts from the Brazilian government to modernize and increase the level of control over the fiscal transactions for all companies. It is based on a digital file that is transmitted periodically to the government via the internet. Similar to the NFe, the file is digitally signed through specific programs that validate its format and content.

The strategy adopted to address this requirement was to keep the transaction features in Oracle Retail Fiscal Module (ORFM) and the interface to the Fiscal Authority in Oracle's fiscal partners.

To support this strategy, views and tables make available all information to the fiscal partner based on the fiscal movements.

SPED File Structure

The SPED file that is generated by the fiscal partners contains a structure organized in blocks with opening and closure registers. The information in each block is as follows:

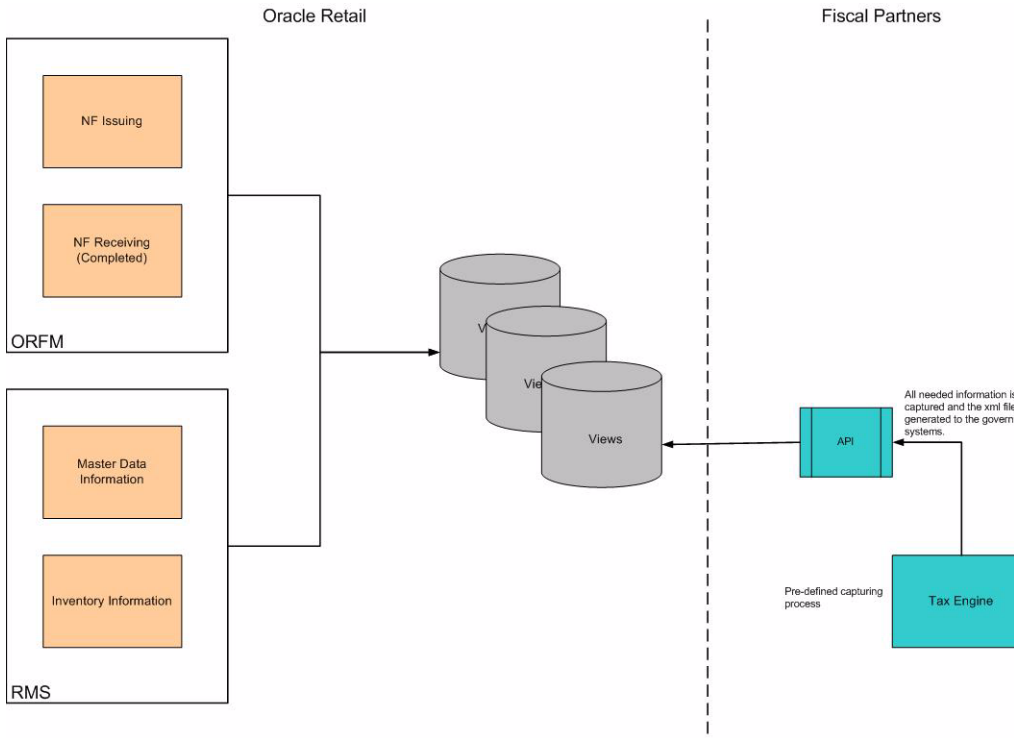
- Block 0: Opening, Identification and References
- Block C: Fiscal Documents I - Merchandise (ICMS/IPI)
- Block D: Fiscal Documents II - Services (ICMS)
- Block E: Fiscal counting of ICMS and IPI
- Block H: Physical Inventory
- Block 1: Other information
- Block 9: Control and Closing of the Digital File

The file is generic and includes information pertinent to all types of companies. The retail segment is required to fill part of the entire file. In addition, the fiscal partner will be in charge of completing the information that is not provided by Oracle Retail.

Overview

The overall solution landscape is based on the existing views that integrate fiscal and master information with Oracle's fiscal partners.

Figure 8–9 SPED Integration



Because the SPED file has several sections corresponding to all types of transactions and fiscal data for a company, the scope of the integration (from the commercial system standpoint) was to make available all data kept within RMS and ORFM. Because of this, all types of data related to products for resale, and the fiscal transactions related to this type of product, are available in ORFM views and tables.

Products used for consumption, assets, and services, and all transactions related to these types of product, are out of the scope for RMS/ORFM and are not available in the views.

The views and tables created to feed SPED include only data available in RMS/ORFM. The file is generated by the fiscal partner´s solution, and fields (such as file opening and file closing), data related to the version of the SPED program, and all specific data for the file is provided by the fiscal partner. In addition, any field that can be deduced by the fiscal partner should also be provided by them.

SPED interfaces with a third-party system that shares the RMS database and opens the ports to establish network connectivity. It depends on the decision of the client to either host the SPED interfacing application (Interdados) within their environment or host it in a fiscal partner's environment. For security considerations, a separate schema should be created that contains only synonyms to as many objects required by the fiscal partner to generate the SPED information. Only the ‘select’ privileges should be granted on these synonyms. No insert/update/delete should be allowed.

9

RMS Advanced Queueing Implementation

Oracle Advanced Queueing (AQ) is the Oracle database queue management feature. AQ provides an API for enqueuing messages to database queues. These messages can later be dequeued for asynchronous processing. Oracle AQ also provides functionality to preserve, track, document, correlate, and query messages in queues.

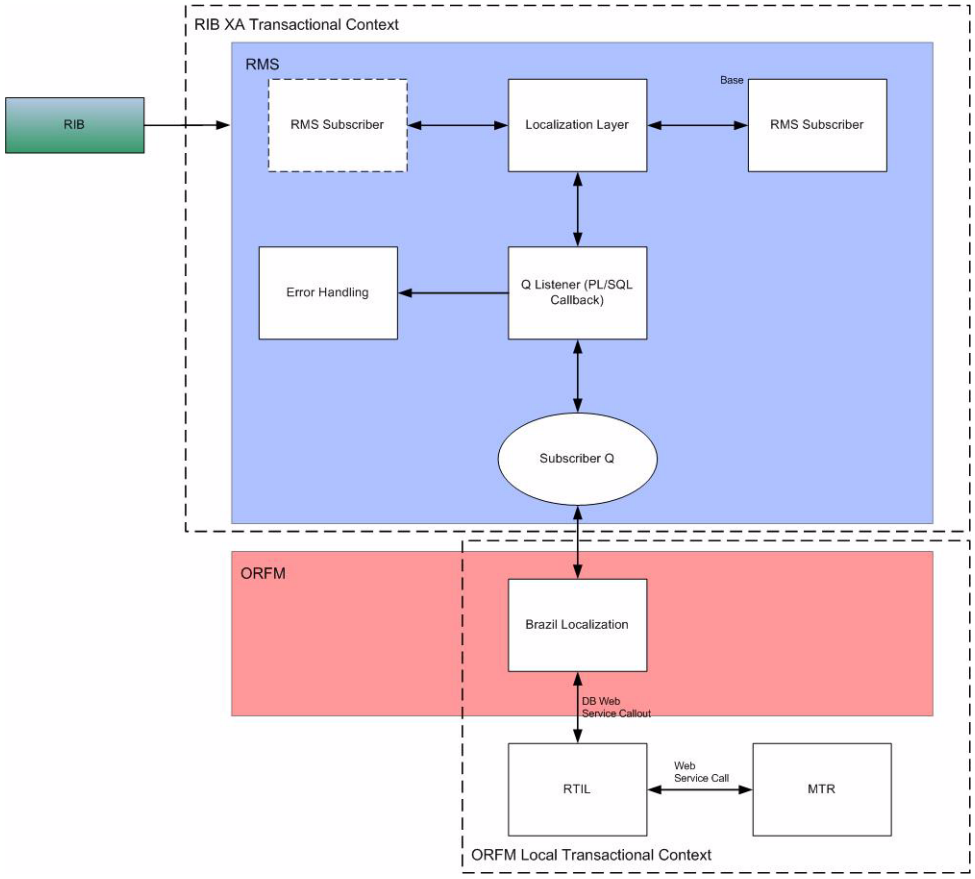
ORFM/RMS integrates with a third-party tax solution provider Mastersaf through web-service interfaces. For certain Xmessages, this call happens within RIB transactional context. From a technical standpoint, invocation of Web services within an XA transactional context is not supported. Therefore, the RIB messages are stored in a temporary (staging) table, and the AQ feature of the Oracle Database is used to facilitate real-time processing of messages. During processing, the RIB message is broken into two transactions:

- RIB transaction is broken at some logical point before any external call is made so as to preserve the sanctity of the XA transaction. This is basically from the point RIB transaction is initiated till enqueue is done.
- Remaining operations of original RIB transactions are carried out in a single transaction. Since this is disconnected from RIB, external system call can be made from within this transaction. This is from dequeue up until the transaction is completed regardless of the number of MTR calls.

The interface with the external taxation solution is only for customers who implement Brazil localization, and the base configuration does not utilize this functionality. This functionality requires a queueing infrastructure to reduce the RIB XA transactional context and ensure there are no violations in invoking Web service calls. Oracle Streams AQ is used as the queuing methodology. Since RIB messages are routed to ORFM through RMS, the AQ/DQ logic is implemented in RMS. Messages that would require Web service calls are enqueued.

Technical Implementation

Figure 9–1 Enqueue/Dequeue Technical Implementation



Enqueue Process Flow (Within RIB Transactional Context)

The entire enqueue process flow works within the RIB XA transactional context. Any failures are handled using the normal RIB error handling infrastructure.

On RIB invocation of the RMS consume function, a check is done to see if the message needs to be enqueued or not. The enqueue decision is based on whether the message requires external Web service calls. If yes, the data is inserted into a staging table and a message log table. The status in the log table is set to "U" indicating it is unprocessed.

If the insertion into the staging table is successful, then enqueue happens immediately with the payload (sequence number of the newly created record and the message family and message type). The message is queued using default options for the enqueue procedure. If this enqueue is successful, a success message is sent back to RIB which commits the transaction.

Dequeue Process Flow (Database Transaction Context)

The entire dequeue process flow works in a single local database transaction context. When RIB commits the transaction, a PL/SQL callback procedure is triggered automatically to dequeue the message. This automatic dequeue happens via notification whenever a message is enqueued. Oracle notifies an agent to execute a registered PL/SQL callback procedure (alternatively, the agent can notify an email address or http:// address rather than execute a callback procedure).

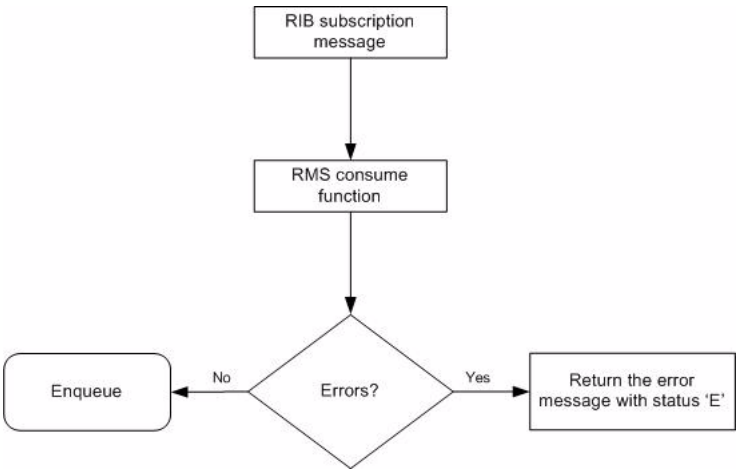
Within this callback procedure, the message log table is consulted to check if there is a record with "E" (erorr) status for the same transaction type and business object ID. If a record is found, the message is directly put into the error log as the prerequisite message needs to be processed first. If not, the callback procedure calls the ORMS base package to start the transaction on its own. A localization decoupling layer will route the message to ORFM, if a localized Brazil location is detected, and call Mastersaf Web services for fiscal management or tax calculation. All these transactions are happening under a single ORMS local transactional context. On successful completion of the processing, the corresponding record in the staging table will be deleted and the message log will be updated to status "S". Failures in any of these operations result in a rollback to the local transactional context and updating the message log to status "E". Error handling is done separately in case if there are any errors occurred during the dequeue process. See the "Error Handling after Dequeue Process" section for additional information.

AQ Error Handling

Error Handling Up to the Enqueue Process

There are no database commits or rollbacks within this flow, and any errors are directed back to RIB. If an error occurs during the dequeue process, the user has to see the error message in the exception queue table. Appropriate action should be taken based on the message.

Figure 9–2 Enqueue Error Handling



Error Handling after Dequeue Process

Error handling after the dequeue process is made up of two components, error capture and error recovery.

Error Capture

Any error that occurs in the RMS or ORFM flows is logged into an error log table. Since the entire dequeue process is operated in a single RMS local transaction context, it is necessary to use autonomous transactions to ensure that the error details are logged into the error log table. This allows errors to be persisted, despite the main transactions undergoing a transaction rollback in case of errors.

The error log table, in conjunction with the message log table and the staging table, are used in the error recovery process.

During the error capture process, the error log entries capture the following attributes:

- TransactionType: This depicts the business transaction type which has an error. It matches with the RIB message family name. (For example: ASNOUT, RECEIVING, RTV, INVADJ, XTSF, XORDER, XITEM and XITEMLOC).
- DocId: This depicts the business document id. (For example: PO number or Transfer Order number, ASN number or RTV number.)
- DocType: This depicts the business document type corresponding to the document id. (For example: PO, TSF, RTV, ASN)
- Item: This depicts the item involved in the transaction.
- LocType: This attribute depicts the location type (Store/Warehouse).

- Location: This attribute depicts the location Id of the transaction which had an error.
- Error Message: This attribute depicts the details of the error. In case of automated retries, this field contains the error message associated with the most current retry.
- AQMsgId: This attribute depicts the enqueued message id.
- CreateTime: This attribute represents the time of error occurrence. Timestamp of when the error was logged.
- CreateID: This attribute represents the user id of the process that logged the error. User id of when the error was logged.
- LastUpdateTime: This attribute represents the last update time of the error log.
- LastUpdateID: This attribute represents the user id of the process that last updated the error log.

Oracle Streams AQ retry feature is used for automatic retries of failed transactions. If the retry count is exceeded, AQ infrastructure moves the message to an exception queue. During queue creation, a default retry count is set.

Error Recovery

The error recovery/handling for all RMS flows (including the exit and entry flows of ORFM) which involve a taxation call is driven through a consolidated error recovery user interface infrastructure.

The error recovery user interface provides the following functionality:

- The appropriate active error records are retrieved from the error log and displayed in the user interface.
- The Error recovery user interface provides the search filters to search the error records by location type, location ID, transaction type, and transaction ID.
- A Reprocess button to trigger the reprocessing of selected records.
- A Delete button to trigger the deletion of selected records. This is so that messages that do not have a chance to be reprocessed successfully and require the source system (e.g. SIM or RWMS) to resend the message again can be removed from the error log.

Reprocessing Error Records

Reprocessing of the selected records occurs in synchronous fashion. In the event of multiple records selected for reprocessing, the processing order will be in the ascending order of the messages logged on to the error log table. In the event of failures in reprocessed transactions, failures are recorded back in the error log table against the same record with update time and updated error message. Successful processing of reprocessed records removes the records from the error log table and the staging table. This also dequeues the messages from the exception queue table as this contains the entries that failed beyond the permissible retry limit.

10

Batch Processes

Batch overviews tie a functional area description to the batch processes illustrated in the designs. The overviews allow the reader to quickly determine how a business function works behind the scenes.

Batch designs describe how, on a technical level, an individual batch module works and the database tables that it affects. In addition, batch designs contain file layout information that is associated with the batch process.

Mastersaf Tax Engine Integration Batch

Batch Design Summary

The following batch designs are included in this functional area:

- FISDNLD.PC (Fiscal Download)
- FRECLASS.PC (Fiscal Reclassification)
- FRECLSPRG.PC (Fiscal Reclassification Purge)

fisdnld (Fiscal Download)

Functional Area

Mastersaf Tax Engine Integration

Module Affected

FISDNLD.PC

Design Overview

This batch downloads the Fiscal attributes from the external tax engine. The attribute can be passed as an input parameter to the program. The attribute is the KEY to be downloaded from the Tax engine and it can be passed as input parameter to the program. No parameter is required, if the user wants to download all the fiscal attributes. The cut off date to download the data from Mastersaf is fetched from FISCAL_ATTR_UPDATE table. This program calls the API, TaxRulesServiceConsumer.GETEXPORTDATA exposed by RTIL (Retail Tax Integration Layer) to the country specific Fiscal attributes.

It loads the foundation data in the respective tables as mentioned below.

NCM_CODES with NCM codes

NCM_CHAR_CODES with NCM Characteistics codes

NCM_IPI_CODES with EX-IPI codes

NCM_PAUTA_CODES with the NCM Pauta codes

FEDERAL_SERVICE_CODES with the Federal service codes

MASTERSAF_SERVICE_CODES with Mastersaf specific codes

CNAE_CODES with the CNAE codes.

Finally, the last_upd_date in FISCAL_ATTRIB_UPDATES table will be updated to the current vdate for the corresponding attributes once the data is loaded.

Scheduling Constraints

Table 10–1

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	This program can run in ad-hoc basis whenever the new fiscal attributes needs to be downloaded from Mastersaf.
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

Key Tables Affected

Table 10–2

Tables	Select	Insert	Update	Delete
NCM_CODES	Yes	Yes	Yes	No
NCM_CHAR_CODES	Yes	Yes	Yes	No
NCM_CHAR_CODES	Yes	Yes	Yes	No
NCM_PAUTA_CODES	Yes	Yes	Yes	No
FEDERAL_SERVICE_CODES	Yes	Yes	Yes	No
MASTERSAF_SERVICE_ CODES	Yes	Yes	Yes	No
CNAE_CODES	Yes	Yes	Yes	No
FISCAL_ATTRIB_UPDATES	Yes	Yes	Yes	No

I/O Specification

N/A

freclass (Fiscal Reclassification)

Functional Area

Mastersaf Tax Engine Integration

Module Affected

FRECLASS.PC

Design Overview

This batch processes all the items scheduled for Fiscal Reclassification changes. The fiscal reclassification information is stored in FISCAL_RECLASS table. It updates the flex field table item_country_l10n_ext using the new fiscal reclassification data available in FISCAL_RECLASS table. The tax engine call happens for the reclassified items and the GTAX_ITEM_ROLLUP table will be updated with the tax details for sale and purchase.

Scheduling Constraints

Table 10–3

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	This program can run in ad-hoc basis.
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	Threading based on reclassification id.

Restart/Recovery

The logical unit of work for this module is defined by vdate and reclassication_id combination. This batch program uses table-based restart/recovery. The commit happens in the database when the commit_max_ctr is reached.

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

Key Tables Affected

Table 10–4

Tables	Select	Insert	Update	Delete
FISCAL_RECLASS	Yes	No	Yes	No
COUNTRY_ATTRIB	Yes	No	No	No
ITEM_COUNTRY_L10N_EXT	No	No	Yes	No
GTAX_ITEM_ROLLUP	Yes	No	Yes	No

I/O Specification

N/A

freclsprg (Fiscal Reclassification Purge)

Functional Area

Mastersaf Tax Engine Integration

Module Affected

FRECLSPRG.PC

Design Overview

This batch purges the processed reclassification data from FISCAL_RECLASS table. The records to be purged are based on its processed_date or active_date less than the current vdate along with the status.

Scheduling Constraints

Table 10–5

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	This program can run after the successful completion of freclass.pc
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	Threading based on reclassification id.

Restart/Recovery
N/A

Locking Strategy
N/A

Security Considerations
N/A

Performance Considerations
N/A

Key Tables Affected

Table 10–6

Tables	Select	Insert	Update	Delete
FISCAL_RECLASS	Yes	No	No	Yes
PERIOD	Yes	No	No	No

I/O Specification
N/A

Financial Postings Batch

Batch Design Summary

The following batch designs are included in this functional area:

- fmfinpost.pc
- fmtrandata.pc

fmfinpost.pc

Functional Area
Financial Postings

Module Affected
fm_financial_posting_sql

Design Overview
Rolling up of transaction amount into accounts based on gl cross ref.

Scheduling Constraints

Table 10–7

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	Ad-hoc
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery
N/A

Key Tables Affected

Table 10–8

Tables	Select	Insert	Update	Delete
fm_gl_options	Yes	No	No	Yes
Fm_coa_setup	Yes	No	No	No
Fm_sob_setup	Yes	No	No	No
Fm_account_setup	Yes	No	No	No
Fm_dynamic_segment_setup	Yes	Yes	No	No
Fm_gl_cross_ref	Yes	Yes	No	No
Fm_gl_dynamic_attributes	Yes	Yes	No	No
Fm_tran_data	Yes	No	No	No
fm_fiscal_doc_header	Yes	No	Yes	No
fm_schedule	Yes	No	Yes	No

I/O Specification
N/A

fmtrandata.pc

Functional Area
Financial Postings

Module Affected
fm_trandata_posting_sql

Design Overview
Computation of the Transaction data based on the Transaction codes.

Scheduling Constraints

Table 10–9

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	Ad-hoc
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery
N/A

Key Tables Affected

Table 10–10

Tables	Select	Insert	Update	Delete
fm_tran_data	No	Yes	No	No
fm_fiscal_doc_header	Yes	No	Yes	No

I/O Specification
N/A

Purging Process Batch

fmpurge.pc

Functional Area

Purging Process

Design Overview

In production environment, as the number of transactions increases over a period of time; in order to keep the performance intact it is required to keep purging the data from the active tables of the application periodically. This batch purges the data from the active tables and stores them in history tables.

Scheduling Constraints

Table 10–11

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	Ad-hoc
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table 10–12

Tables	Select	Insert	Update	Delete
fm_schedule_hist	No	Yes	No	No
fm_fiscal_doc_header_hist	No	Yes	No	No
fm_fiscal_doc_detail_hist	No	Yes	No	No
fm_tran_data_hist	No	Yes	No	No
fm_receiving_header_hist	No	Yes	No	No
fm_receiving_header_hist	No	Yes	No	No
fm_receiving_detail_hist	No	Yes	No	No
fm_fiscal_doc_tax_head_hist	No	Yes	No	No

Table 10–12

Tables	Select	Insert	Update	Delete
fm_fiscal_doc_tax_detail_hist	No	Yes	No	No
fm_resolution_hist	No	Yes	No	No
fm_correction_tax_doc_hist	No	Yes	No	No
fm_correction_doc_hist	No	Yes	No	No
fm_nf_doc_tax_head_ext_hist	No	Yes	No	No
fm_nf_doc_tax_detail_ext_hist	No	Yes	No	No
fm_nf_doc_tax_detail_wac_hist	No	Yes	No	No
fm_fiscal_doc_payments_hist	No	Yes	No	No
fm_nf_doc_tax_rule_ext_hist	No	Yes	No	No
fm_sped_fiscal_doc_header_hist	No	Yes	No	No
fm_sped_fiscal_doc_detail_hist	No	Yes	No	No
fm_schedule	Yes	No	No	Yes
fm_fiscal_doc_header	Yes	No	No	Yes
fm_fiscal_doc_detail	Yes	No	No	Yes
fm_tran_data	Yes	No	No	Yes
fm_receiving_header	Yes	No	No	Yes
fm_receiving_header	Yes	No	No	Yes
fm_receiving_detail	Yes	No	No	Yes
fm_fiscal_doc_tax_head	Yes	No	No	Yes
fm_fiscal_doc_tax_detail	Yes	No	No	Yes
fm_resolution	Yes	No	No	Yes
fm_correction_tax_doc	Yes	No	No	Yes
fm_correction_doc	Yes	No	No	Yes
fm_fiscal_doc_tax_head_ext	Yes	No	No	Yes
fm_fiscal_doc_tax_detail_ext	Yes	No	No	Yes
fm_fiscal_doc_tax_detail_wac	Yes	No	No	Yes
fm_fiscal_doc_payments	Yes	No	No	Yes

Table 10–12

Tables	Select	Insert	Update	Delete
fm_fiscal_doc_tax_rule_ext	Yes	No	No	Yes
fm_sped_fiscal_doc_header	Yes	No	No	Yes
fm_sped_fiscal_doc_detail	Yes	No	No	Yes

I/O Specification
N/A

Localization Batch

refmvl10entity (Refresh MV MV_L10N_ENTITY)

Functional Area
N/A

Module Affected
REFMVL10ENTITY.PC

Design Overview
This is an ad hoc batch program that refreshes the materialized view MV_L10N_ENTITY that is based on ADDR, OUTLOC, COMPHEAD, COUNTRY_ATTRIB table.

Scheduling Constraints

Table 10–13

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery
This batch program uses table-based restart/recovery. The commit happens in the database when the commit_max_ctr is reached.

Locking Strategy
N/A

Security Considerations
N/A

Performance Considerations
N/A

Key Tables Affected

Table 10–14

Tables	Select	Insert	Update	Delete
ADDR	Yes	No	No	No
OUTLOC	Yes	No	No	No
COMPHEAD	Yes	No	No	No
COUNTRY_ATTRIB	Yes	No	No	No

I/O Specification
N/A

SPED Batch

Import_sped

Functional Area
SPED

Module Affected
Import_SPED.ksh

Design Overview
This batch will insert all processed NFs (status = 'FP') into two tables for SPED - FM_SPED_FISCAL_DOC_HEADER and FM_SPED_FISCAL_DOC_DETAIL. It looks into the driving table - FM_SPED_LAST_RUN_DATE for the last run date of SPED, to fetch all 'C'losed NFs whose transaction amounts are also rolled up into ledger accounts based on gl_cross_ref in between last run date and sysdate from the main tables of ORFM. Once the records are successfully inserted, the batch updates the last _run_date column of the driving table - FM_SPED_LAST_RUN_DATE to sysdate.

It is recommended to run this batch job on a daily basis due to performance impacts. This batch has a pre-dependency on the ORFM Financial postings batch - FMFINPOST.PC , so once Financial postings batch program completes and sets the Nota Fiscal status to 'F'inancially 'P'osted (FP) only after that SPED insert batch job - import_SPED.ksh should be triggered to fetch all such 'F'inancially 'P'osted (FP) NFs from the ORFM tables.

Scheduling Constraints

Table 10–15

Schedule Information	Description
Processing Cycle	Ad-hoc
Scheduling Considerations	This program should run only after the successful completion of FMFINPOST.PC .
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

Key Tables Affected

Table 10–16

Tables	Select	Insert	Update	Delete
FM_FISCAL_DOC_ HEADER	Yes	No	No	No
FM_FISCAL_DOC_ DETAIL	Yes	No	No	No
FM_SCHEDULE	Yes	No	No	No
FM_FISCAL_DOC_TAX_ HEAD	Yes	No	No	No
FM_FISCAL_DOC_TAX_ DETAIL	Yes	No	No	No
FM_FISCAL_DOC_TAX_ DETAIL_EXT	Yes	No	No	No
FM_FISCAL_ UTILIZATION	Yes	No	No	No
FM_FISCAL_DOC_ PAYMENTS	Yes	No	No	No
FM_SPED_FISCAL_DOC_ HEADER	No	Yes	No	No

Table 10–16

Tables	Select	Insert	Update	Delete
FM_SPED_FISCAL_DOC_DETAIL	No	Yes	No	No
ITEM_MASTER	Yes	No	No	No
FM_SPED_LAST_RUN_DATE	Yes	No	Yes	No
V_BR_ITEM_FISCAL_ATTRIB	Yes	No	No	No
V_FISCAL_ATTRIBUTES	Yes	No	No	No

I/O Specification

N/A