Retail passion. Innovative solutions.

Retek® Customer Order Management™ 11.0

Operations Guide

Corporate Headquarters:

Retek Inc.
Retek on the Mall
950 Nicollet Mall
Minneapolis, MN 55403
USA

888.61.RETEK (toll free US)
Switchboard:
+1 612 587 5000

Fax:
+1 612 587 5100

European Headquarters:

Retek
110 Wigmore Street
London
W1U 3RW
United Kingdom

Switchboard:
+44 (0)20 7563 4600

Sales Enquiries:
+44 (0)20 7563 46 46

Fax:
+44 (0)20 7563 46 10

Customer Support

Customer Support hours

Customer Support is available 7x24x365 via email, phone, and Web access.

Depending on the Support option chosen by a particular client (Standard, Plus, or Premium), the times that certain services are delivered may be restricted. Severity 1 (Critical) issues are addressed on a 7x24 basis and receive continuous attention until resolved, for all clients on active maintenance. Retek customers on active maintenance agreements may contact a global Customer Support representative in accordance with contract terms in one of the following ways.

Contact Method	Contact Information
E-mail	support@retek.com
Internet (ROCS)	rocs.retek.com Retek's secure client Web site to update and view issues
Phone	+1 612 587 5800

Toll free alternatives are also available in various regions of the world:

Australia	+1 800 555 923 (AU-Telstra) or +1 800 000 562 (AU-Optus)
France	0800 90 91 66
Hong Kong	800 96 4262
Korea	00 308 13 1342
United Kingdom	0800 917 2863
United States	+1 800 61 RETEK or 800 617 3835
Mail	Retek Customer Support Retek on the Mall 950 Nicollet Mall Minneapolis, MN 55403

When contacting Customer Support, please provide:

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

Contents

Chapter 1 – Introduction

This operations guide serves as a Retek Customer Order Management (RCOM) reference to explain 'backend' processes, including batch processing. RCOM is designed as a standalone enterprise order management application.

To enable enterprise order management, RCOM has been architected on a J2EE Java architecture, which facilitates an effective integration of order management system (OMS) functions into external applications and provides for the management of common business logic across the enterprise.

Overview

RCOM is the primary system that combines your customers, merchandise, promotions, and pricing to affect the customer's shopping experience. As a result, it is the primary system that governs how customers and prospective customers are treated as they interact with a retailer for purchases or product information. The system is Retek's high-speed, high-volume, and multi-channel consumer order management system that unifies state-of-the-art order-entry, real-time available to promise (ATP) processing, product information, customer service, and fulfillment functions.

Order management becomes very complex as customers, prospects, merchandise, and promotions are managed across multiple channels and banners. This release of RCOM is designed with the flexibility to account for this complexity.

RCOM's functionality includes complete visibility of the customer order lifecycle, from order capture to fulfillment to post customer service activities. Retek software provides this visibility with unified integration of state-of-the-art order entry, customer service, fulfillment, and sales processing functions.

The RCOM application is designed and built as a dedicated business-to-consumer order management solution to provide comprehensive order capture, order management and order fulfillment of customer orders.

Retailers benefit from RCOM's J2EE distributed computing platform that offers:

- Interaction among layers provided through public interfaces.
- Transaction management through defined transactional boundaries.
- Distributed implementation, with some layers in the model able to perform in independent locations.
- Security authentication functionality.
- Java database connectivity (JDBC) in the DAO layer, minimizing the number of interface points that need to be maintained.

Who this guide is written for

Anyone who has an interest in better understanding the inner workings of the RCOM system can find valuable information in this guide. There are two audiences in general for whom this guide is written:

- System analysts who are looking for information about RCOM's processes internally or in relation to the systems across the enterprise.
- System operation personnel who operate RCOM on a regular basis.

Where you can find more information

- RCOM front-end documentation (for example, the RCOM User Guide)
- RCOM Integration Guide
- RCOM Installation Guide
- Retek Warehouse Management System (RWMS) product documentation
- Retek Merchandising System (RMS) product documentation
- Retek Integration Guide and other RIB-related documentation
- Applicable third-party documentation (such as for Vertex, and so on)

Chapter 2 – Backend system administration and configuration

This chapter of the operations guide is intended for database administrators who provide database support and monitor the running system.

The content in this chapter is not procedural, but is meant to provide descriptive overviews of key system configuration parameters and tools.

Supported Retek products

This version of RCOM is compatible with the following Retek products:

- RMS 10.2.2
- RIB 11.0.1
- RDM 10.3.5

Supported environments

This version of RCOM has been certified on the following platform, with the following components:

- Operating System: AIX v5.1 or v5.2
- Java Virtual Machine (Windows): J2RE 1.4.2 IBM Windows 32 build cn131-20030618
- Java Virtual Machine (AIX): J2RE 1.4.2 IBM AIX build ca131-20030618
- Java Web Start installed from 1.4.2 JRE
- Application Server:
  - IBM WebSphere Application Server 5.1.1
  - IBM WebSphere Application Network Deployment 5.1.1 (optional)
  - JDBC Driver: Oracle JDBC Driver 9.2.0.3
- Database: Oracle 9.2.0.4
- Vertex 2.1.4
- Microsoft Active Directory MS2000 - 5.2.3790

Recommended RCOM client system requirements

The following list describes the minimum requirements that are necessary to run this version of RCOM:

- Memory – 512 MB
- CPU – 1 GHz or faster
- Screen Resolution – 1024x768

Exception handling

The two primary types of exceptions within the RCOM system are the following:

- System exceptions
  For example, server connection and/or database issues are system exceptions.
- Business exceptions
  For example, mandatory information that is not included in a customer address results in a business exception's being thrown. Most exceptions that arise in the system are business exceptions.

Unwrapping business exceptions

The RCOM business exception model is such that the business exception itself is an object that contains problems. For example, during validation, an order's creation might result in five validation errors. For each error, a problem is created that is added to the business exception. The user interface can then interpret the 'problems' and display them.

The screen capture below illustrates the business exceptions within RCOM. It is associated to the following location in the Javadoc:

- com.retek.commons.domain.core.exception.BusinessException

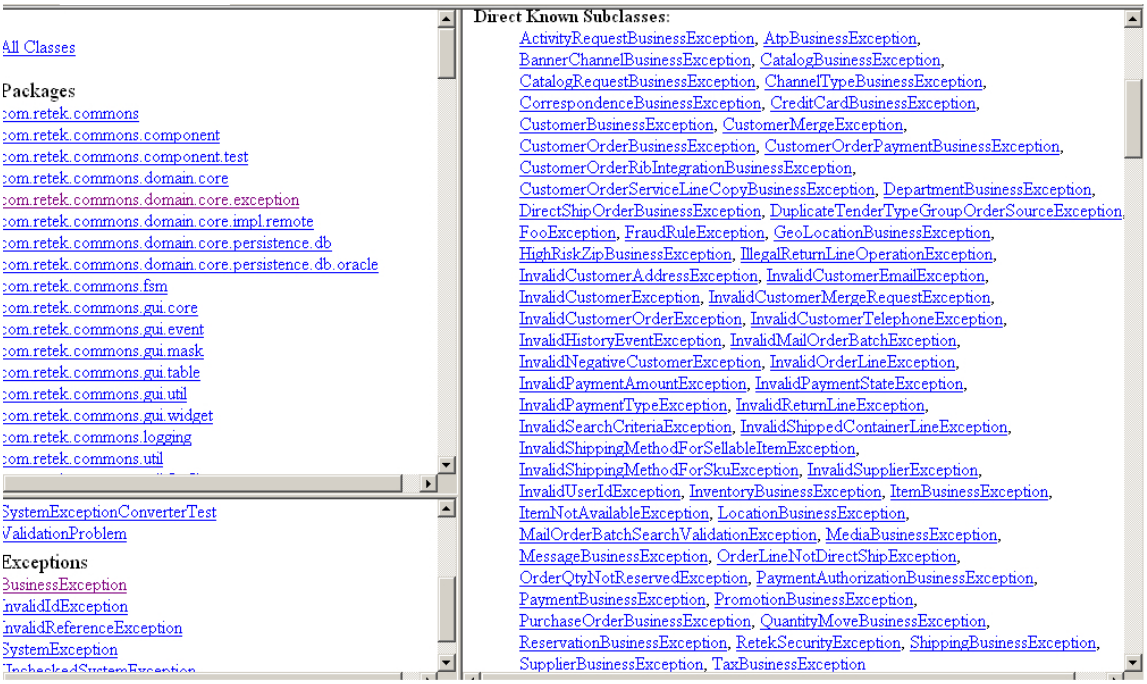
When the system is finished validating, the user can identify the business-specific exception. Once the exception is identified, the developer can determine or 'get' the problems through these methods:

```
getProblems
```

Or

```
hasProblems
```

Once a collection of the problems is returned, the developer can determine the problems and their severity.



Logging standards

Logging pattern

As the example below illustrates, logging within RCOM follows the pattern established in the properties file, log4j.properties. All of the logs in the system conform to this format, which addresses when, where, what and why.

The example below includes the following data:

- The date and time, including time in milliseconds
- The level
- The class that writes to the log (for example, a batch class that caught an exception and logged the error)
- The message

```
Example:
2003-06-18 11:06:25,684 FATAL [TestClass] : This is a fatal log message.
2003-06-18 11:06:25,694 ERROR [TestClass] : This is an error log message.
2003-06-18 11:06:25,694 INFO  [TestClass] : This is an info log message.
2003-06-18 11:06:25,694 DEBUG [TestClass] : This is a debug log message.
2003-06-18 11:06:25,694 TIMING  [TestClass] : This is a timing log message.
10:36:17,193 TIMING [TestServicesEjb] :
<- com.package.TestServicesRemoteHome.create() [5ms]
```

Logging levels

The logging mechanism that is used for RCOM is log4j, which is the same as the server's flat text log file. This logging mechanism reveals errors and other significant events that occur during the system's runtime processing, including its batch processes. In most cases, business exceptions 'rise' to the user interface. If a business exception is displayed, it is logged. Log4j is an open source product.

By adding a 'Timing' level, Retek's logger extends the log4j logging levels. When the system is set to log at a 'Timing' logging level, the system records the time of any Enterprise Java Bean (EJB) method call that occurs. When the code needs to create an EJB and the timing logging level is set, the system creates layers of wrapper around the EJB. When calls are made to the EJB, the wrappers facilitate the creation of the timing data. For example, when an order is submitted through a method, the system could log that action as taking 7 milliseconds.

Note: In a production environment, the logging setting should be set to Error or Fatal so that system performance is not adversely impacted.

The level setting established in the properties file, log4j.properties, instructs the system to log that level of error and errors above that level.

The logging levels are the following:

- Fatal
- Error
- Warn
- Info
- Debug
- Timing

Log4j properties file and logging levels

In the log4j properties file, a setting establishes the root logger.

The loggers defined in the application extend from the root logger. Logging can be set at any level within the application. For example, logging can be set at the component level, the class level, and so on. This level of logging can be helpful when troubleshooting specific parts of the application. For example, if the application is experiencing issues in a specific area such as a customer order line, the logger could be set to debug.

If no level has been defined for a specific level, the root logger applies.

Additional information about log4j and its logging levels can be found at the following website:

- http://jakarta.apache.org/log4j/docs/index.html

Summary of RCOM events and logging levels

The following table illustrates the relationship between RCOM events and their associated logging levels. A description of the type of information recorded is also included.

RCOM event	Logging level	Description
SystemException	Fatal	Stack Trace. Information about the business process that failed. Information about the method that failed.
RuntimeException	Fatal	Stack Trace. Information about the business process that failed. Information about the method that failed.
BusinessException	Error	Stack Trace. Information about the business process that failed. Information about the method that failed.
EJB method timings	Timing	Method name, method execution time
SQL dump in all DbOperation classes	Debug	SQL statement
Debug Event	Debug	Relevant message intended for developer debugging
Batch program runtime information	Info	Relevant progress messages
RIB publish/subscribe runtime information	Info	The injector/publisher name. The XML stream
Swallowed Exceptions	Info	Stack Trace. Information about the business process that failed. Information about the method that failed.

‘Metal’ look and feel parameter in COM properties

RCOM can either have the ‘look and feel’ of its GUI specified, or it can rely on the operating system. If a retailer is running on Windows XP, the retailer must set the parameter within the COM properties file to ‘`metal`’, which is the standard Java look and feel. If this parameter is not set, the Windows XP Operating System does *not* successfully handle the ‘look and feel’ of the GUI.

Sandbox sizing requirements

Backend

- Operating System: AIX v5.1 or v5.2
- Database: Oracle 9.2.0.4
- System: P570

Note: The values below apply to a small environment.

  - 2 Gig of RAM
  - 2 processors
  - 100 GB of disk space

Middle tier

- Operating System: AIX v5.1 or v5.2
- App Server: Websphere (5.1.1 or 5.2)
- IBM WebSphere Application Net Deployment (5.1.1 or 5.2) (optional)
- JDK/JRE: IBM 1.4.2
- JDBC Driver: Oracle JDBC Driver 9.2.0.3
- System: P570

Note: The values below apply to a small environment.

  - 2 GB of RAM
  - 2 processors
  - 100 GB of disk space

Client

- Operating System: Windows 2000 or XP
- JDK/JRE: IBM 1.4.2
- Screen Resolution: 1024x768
- System:
  - 1 GB RAM
  - 1.5 Ghz
  - 10GB HD

Additional components

- User Authentication: Active Directory (MS2000 - 5.2.3790)
- Java Web Start installed from 1.4.2 JRE

Notes

- If the backend and the middle tier are on the same machine, use the values below:
  - System: P570

Note: The values below apply to a small environment.

- 4 Gig of RAM
- 2 processors
- 100 GB of disk space

- User authentication: OpenLDAP or simple file authentication can replace Active Directory as authentication method if needed.
- The environment above for the client is the most favorable. However, it can be replaced with the following values, 500 MB RAM, 1 GHz CPU, 4 GB HD.
- Because RCOM is a Java application, it slows down considerably when one or more other Java applications are running (for example, SIM, RMM, and so on).

Configuring INV_INVENTORY_QUANTITY_CONFIG

The ATP module is responsible for determining inventory quantity-related calculations. This section discusses the configuration of the ATP module's values.

Note: Back order reservations cannot exist in a Reserve bucket; they must only exist in the Future bucket. RCOM's back order release processing does not function if this business rule is not followed.

How INV_INVENTORY_QUANTITY_CONFIG is used

To calculate the ATP value, RCOM uses a hard-coded equation. This equation is the following:

- ATP=Stock–Reserved+Future availability.

By entering values in the table, INV_INVENTORY_QUANTITY_CONFIG, the client configures the inventory 'quantities' that constitute each of the hard-coded buckets described in the equation above. What the client enters into the table determines which inventory 'quantities' are used and whether they are added to or subtracted from each other.

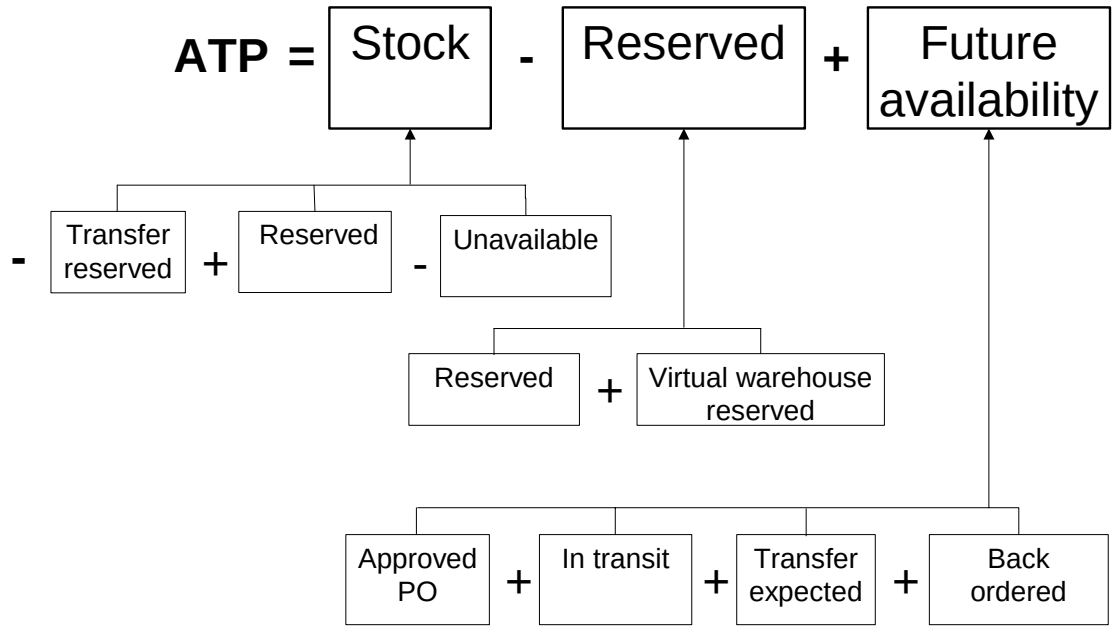
The lower rows in the diagram (following the tables below) illustrate the configurable merchandising inventory quantities that make up the values of the buckets in the hard-coded equation. Note that, except for the top row, all of the buckets shown in the diagram below could be configured differently. Thus, the diagram below shows only one possible configuration, which serves as an example.

For an example, in the INV_INVENTORY_QUANTITY_CONFIG table below, look at the configurable values that are used for the 'Stock' portion (in the inventory_bucket_type_code column) of the hard-coded equation. (If necessary, refer to the Description of possible values for the table INV_INVENTORY_QUANTITY_CONFIG also below.) These bullet points express how the 'Stock' bucket is configured in this one case, which is also demonstrated by the diagram below the tables.

- The ‘Transfer reserved’ bucket is in use (because of the ‘1’ in the included_flag column) but its value is subtracted from the ‘Stock’ bucket (because of the ‘0’ in the added_flag column).
- The ‘Stock on hand’ bucket is in use (because of the ‘1’ in the included_flag column) and its value is added to the ‘Stock’ bucket (because of the ‘1’ in the added_flag column).
- The ‘Unavailable’ bucket is in use (because of the ‘1’ in the included_flag column) but its value is subtracted from the ‘Stock’ bucket (because of the ‘0’ in the added_flag column).

INV_INVENTORY_QUANTITY_CONFIG table			
inventory_qty_type_code	**inventory_bucket_type_code**	**included_flag**	**added_flag**
T	S	1	0
R	R	1	1
B	F	1	0
V	R	1	1
A	F	1	1
I	F	1	1
E	F	1	1
O	S	1	1
U	S	1	0

Description of possible values for INV_INVENTORY_QUANTITY_CONFIG			
inventory_qty_type_code	**inventory_bucket_type_code**	**included_flag**	**added_flag**
T=Transfer reserved	S=Stock	1=true	1=true
R=Reserved	R=Reserved	0=false	0=false
B=Backordered	F=Future available		
V=Virtual warehouse reserved			
A=Approved purchase order (PO)			
I=In transit			
E=Transfer expected			
O=Stock on hand			
U=Unavailable			



An example of configured inventory quantities

Note: Although the inventory quantities are configurable, a client cannot add a new inventory quantity without changing the code. Retek does not recommend this change.

Chapter 3 – Java batch processes

This chapter provides the following:

- An overview of RCOM's batch processing
- A description of how to run batch processes, along with key parameters
- A functional summary of each batch process
- A diagram of dependencies
- A description of some of the features of the batch processes (for example, batch return values)

RCOM's Java batch processes overview

Java batch process architectural overview

The goal of many of RCOM's Java batch processes is to select business objects from the persisted mechanism (for example, a database) by a certain criteria and then to transform them by their state. These RCOM Java-based batch processes remove some of the processing load from the real-time online system and are run periodically. They reside within the component on whose data they work.

To ensure the integrity of the system's transaction management, the batch architecture has been constructed in such as way as to take advantage of the application server's EJB context.

1 By running a finder against the persistence mechanism by a certain criteria, the RcomAbstractBatchProgram class loads the business objects out that need to be processed.

2 Once the batch objects are loaded out, they are passed down to a method that is (in most cases) a command that resides on the application server .

3 The application server processes all of the orders.

Some RCOM Java-based batch processing is file-based. Batch file layout specifications are provided in Appendix A.

Details about the export to the sales audit system is provided later in this chapter and in "Chapter 4 – The RCOM export to a sales audit system (such as ReSA)".

Running a Java-based batch process

Java processes are scheduled through hard-coded executable shell scripts (.sh files). Retek provides each of these shell scripts, which performs the following internally:

- sets up the Java runtime environment before the Java process is run.
- triggers the Java batch process.

For example,

```
./backorderNotification.sh
```

Scheduler and the command line

If the client uses a scheduler, arguments are placed into the scheduler.

If the client does not use a scheduler, arguments must be passed in at the Unix command line.

Java packages and their main class

The following table describes the executable shell scripts and Java packages along with the main class within them that defines the (batch) Java class that runs.

Executable shell scripts	Java package	Main class
backorderNotification.sh	com.retek.component.customerorder.batch	BackorderNotificationBatch
cancelPendedOrder.sh	com.retek.component.customerorder.batch	CancelPendedOrderBatch
catalogRequest.sh	com.retek.component.customer.integration.catalog	CatalogRequestBatch
customerFileExport.sh	com.retek.component.customer.integration.batch	CustomerFileExportBatch
customerFileImport.sh	com.retek.component.customer.integration.batch	CustomerFileImportBatch
customerMergeExport.sh	com.retek.component.customer.integration.batch	CustomerMergeExportBatch
customerMergeImport.sh	com.retek.component.customer.integration.batch	CustomerMergeImportBatch
internetMediaExportBatch.sh **Note:** The command line parameter following the batch process name must be the media number that is being exported. For example, `InternetMediaExportBatch.sh 001` where 001 is the media display code	com.retek.component.internet.batch	InternetMediaExportBatch
masterCreditCardAuth.sh **Note:** Subprocesses for this process are shown in the 'Main class' column of this table.	com.retek.component.customerorder.batch	MasterCreditCardAuthBatch • CreditCardDirectShipReAuthorizationSubprocess • CreditCardNonDirectShipReAuthorizationSubprocess • CreditCardExpirationAndAuthorizationSubprocess

Executable shell scripts	Java package	Main class
mediaDemandUpdate.sh	com.retek.component.demand.batch	MediaDemandUpdateBatch
paymentSettlement.sh	com.retek.component.customerorder.integration.settlement	PaymentSettlementBatch
publishCorrespondence.sh	com.retek.component.customerorder.batch	PublishCorrespondenceBatch
purgeDailyMessageBatch.sh	com.retek.component.customerorder.batch	PurgeDailyMessageBatch
RecalculateBackorderLineECDDBatch.sh	com.retek.component.customerorder.batch	RecalculateBackorderLineECDDBatch
releaseBackorderedLines.sh	com.retek.component.customerorder.batch	ReleaseBackorderedLinesBatch
releaseOrderToWms.sh	com.retek.component.customerorder.batch	ReleaseOrderToWmsBatch
resaRtlogTransformer.sh	com.retek.component.salesaudit.integration.resa	ResaRtlogTransformerBatch
salesAuditExport.sh	com.retek.component.salesaudit.batch	SalesAuditExportBatch
securityUserUpdate.sh	com.retek.component.security.batch	SecurityUserUpdateBatch
updateMediaStatus.sh	com.retek.component.media.batch	UpdateMediaStatusBatch

Functional descriptions

The following table summarizes RCOM's batch processes and includes a description of each batch process's business functionality.

Batch processes	Details
BackorderNotificationBatch	This batch process looks at all order lines in backorder status and determines if a backorder notification needs to be sent to the customer. The batch process inserts into a staging table with the pertinent notification information (order line, template name, delivery method) that needs to be available for inventory management to review before they are sent to the customer.
CancelPendedOrderBatch	Some orders have a form of payment that is associated to an authorization. These orders are pended if their authorization expires. The CreditCardAuthorizationBatch tries to reauthorize payments. If this reauthorization does not occur, CancelPendedOrderBatch cancels orders that have been pending for a certain amount of time.
CatalogRequestBatch	This batch process takes all catalog requests captured by RCOM and formats them to a standard third party format. The data is written to a flat file.
CustomerFileExportBatch	The batch process allows the system to mass export customer information to an external system while maintaining a high level of performance. The customer export batch process is comprised of the following steps: • Find customers ready for export This sub-process finds a set of customer references which match the specified export criteria. • Export customer data to XML This sub-process reads segments of customer data from the customer database (based on the customer references found in step 1) and exports them to an XML file. See "Appendix A – Batch file layout specifications".

Batch processes	Details
CustomerFileImportBatch	The batch process allows the system to import mass amounts of customer updates from an external system. This batch process is comprised of the following steps: • Import customer updates (or adds) This sub-process imports a set of customer import requests from an XML input file. For each customer import request in the file, an entry is written to the CST_CUSTOMER_IMPORT_REQUEST staging table. • Process all pending customer import requests. This sub-process finds all import requests from the CST_CUSTOMER_IMPORT_REQUEST staging table. For each request, the batch process performs the customer merge (and removes the request from the staging table). See “Appendix A – Batch file layout specifications”.
CustomerMergeExportBatch	The batch process allows the system to export merge requests. See “Appendix A – Batch file layout specifications”.

Batch processes	Details
CustomerMergeImportBatch	The batch process allows the system to import merge requests. This batch process is comprised of the following steps: • Import customer merge requests. This sub-process imports a set of customer merge requests from an XML input file. For each customer merge request in the file an entry is written to the CST_CUSTOMER_MERGE_REQUEST staging table. • Process all pending customer merge requests. This sub-process finds all import requests from the CST_CUSTOMER_MERGE_REQUEST staging table, for each request, the batch process performs the customer merge and removes the request from the staging table. See “Appendix A – Batch file layout specifications”.
InternetMediaExportBatch	This batch process exports a media and all of its items to an XML file. The batch process uses a schema file. The purpose of the batch process is to improve performance by minimizing the interactivity between the custom user interface (such as the internet) and the RCOM system.

Batch processes	Details
MasterCreditCardAuthBatch • CreditCardDirectShipReAuthorizationBatch • CreditCardNonDirectShipReAuthorizationBatch • CreditCardExpirationAndAuthorizationBatch	The MasterCreditCardAuthBatch process is a consolidation of three credit card-related batch processes. These three batch processes still exist with their business logic intact, but they all three are triggered by the MasterCreditCardAuthBatch process. • CreditCardDirectShipReAuthorizationBatch Reauthorizes orders a certain number of days before the estimated ship date (direct ship) The batch process reauthorizes credit card payments in 'Expired' status or payments that were initially authorized for $1. • CreditCardNonDirectShipReAuthorizationBatch Reauthorizes orders a certain number of days before the release date (non-direct ship). The batch process reauthorizes credit card payments in 'Expired' status or payments that were initially authorized for $1. • CreditCardExpirationAndAuthorizationBatch This batch process looks at all credit card payment lines and determines if the credit card authorization has expired or validated. If expired, the batch process reauthorizes the payment line if today's date is 'n' days away from an associated orderliness ship date. If validated, the batch process authorizes the payment line if today's date is 'n' days away from an associated orderliness release date.
MediaDemandUpdateBatch	This batch process is run hourly and takes information from a staging table to gather demand information for display. Information can be viewed at the following levels: LTD (Life to Date), WTD (Week to Date), DTD (Day to Date). The process runs for all 24 hours.

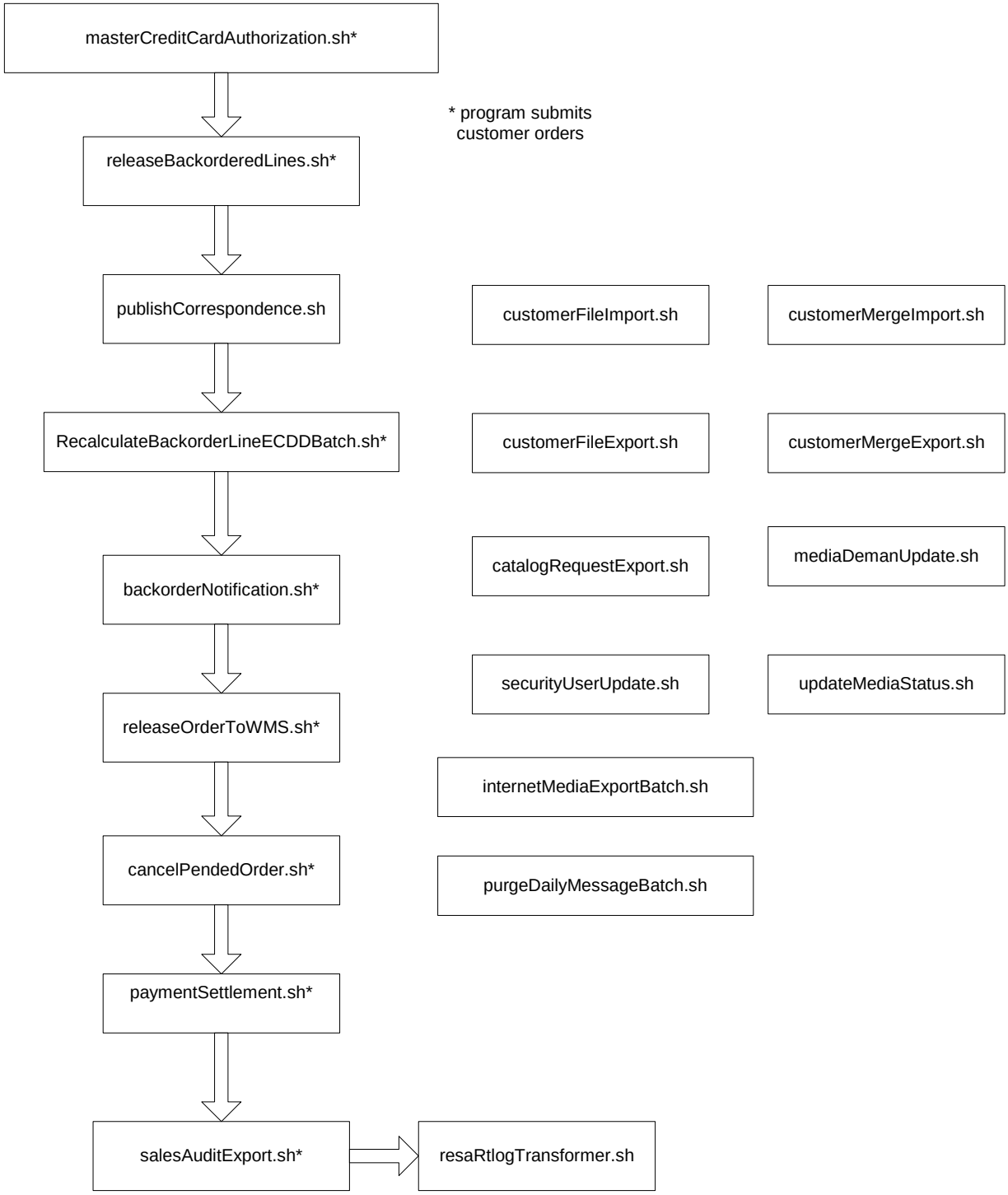
Batch processes	Details
PaymentSettlementBatch	This batch process settles for the amount that was shipped or settles for the amount that was returned. The settlement process determines what has shipped or been returned for a customer order and settles on the appropriate amount related to the transaction. For shipments, the customer is charged once for the value of the merchandise, shipping costs, taxes and value added services. For returns, the customer is refunded merchandise and taxes for the merchandise. The rest of the values are determined whether to be refunded based on the return reason in the system. The batch process consolidates payments, either charged or refunded for a given day, into a single payment. Once the consolidation is complete, the batch process performs one of the following: • Produces records in the credit card settlement file for credit card charges and/or credits. • Publishes a RIB message for physical tender refunds (merchandise vouchers and checks).
PublishCorrespondenceBatch	This batch process takes the notification information from the staging table and publishes the notification.
PurgeDailyMessageBatch	This batch process purges daily messages with a purge date equal to or prior to a user specified date.
RecalculateBackorderLineECDDBatch	This batch process recalculates all backorder line's ECDDs based on items on the staging table COR_ECDD_RECALCULATION_ITEM. The staging table is populated through RIB when there are any changes to purchase orders in the merchandizing system.
ReleaseBackorderedLinesBatch	This batch process attempts to reserve back order quantities.

Batch processes	Details
ReleaseOrderToWmsBatch	This batch process takes all the non-direct ship order lines in reserved status and rolls them up to a ship request (orderlines being sent to the same ship to via the same ship method) and publishes them to the RIB. The WMS takes them from the RIB for fulfillment.
ResaRtlogTransformerBatch	This utility batch process transforms one or more RCOM sales audit export files into ReSA RTLOG format. This program can be used to integrate to ReSA for sales audit.
SalesAuditExportBatch	This batch process extracts key sales information and payment liability transactions from the applicable business objects triggered in the system. For example, shipped containers trigger sales transactions, returns trigger return transactions; an overpayment with a merchandise voucher triggers a paidout for the refund, and so on. The batch process then formats the data into standard generic RCOM XML format, and creates an XML file. See “Appendix A – Batch file layout specifications”.
SecurityUserUpdateBatch	This batch process runs and pulls new and/or modified user-related data from Active Directory and persists the data within RCOM (thus ensuring that the two systems are in sync). The user’s address information is the call center ID. API method calls verify that the call center that is imported from Active Directory is a valid call center in the RCOM system. If the data is not valid, the user’s data is not submitted to RCOM. Rather, the data is written to an output file, which can be specified as an argument in the command line when the batch process is run.
UpdateMediaStatusBatch	This batch process takes all un-active media and verifies that active date is today or before and that all required information is populated. If validation succeeds, the batch process updates the status to active.

Java batch process scheduling flows

Before setting up an RCOM process schedule, familiarize yourself with the scheduling dependencies below.

Note: A process higher in the diagram below precedes a process lower in a diagram.



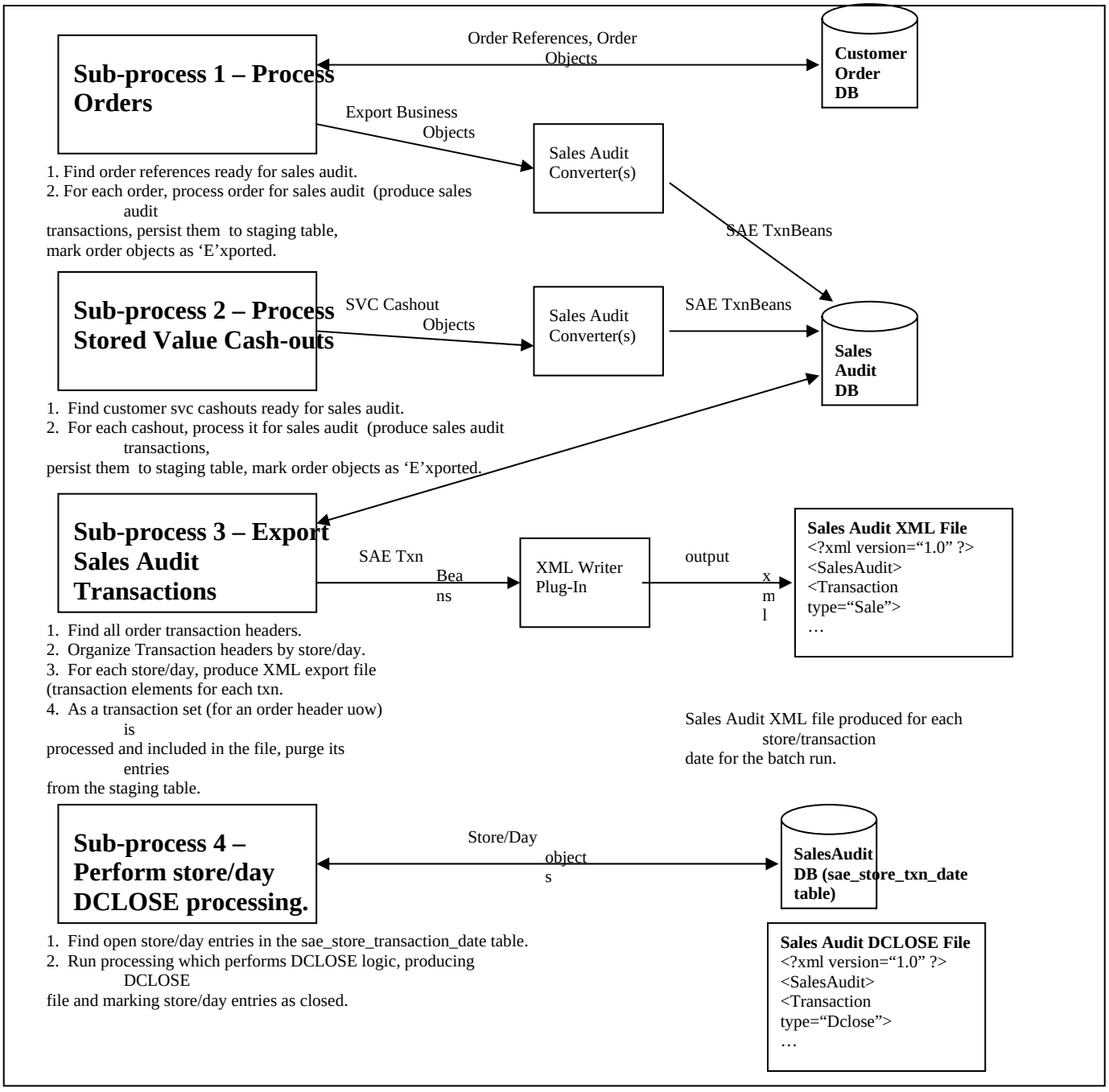
Additional detail about InternetMediaExportBatch

This batch program must include

Additional detail about SalesAuditExportBatch

Sales audit export batch process flow

The following diagram depicts the basic process flow for the batch process that supports the Sales Audit XML export.



Sales audit internal tables

The following tables are used within the sales audit batch processing:

RCOM Table	Description
SAE_STORE_TRANSACTION_DATE	Persistence store for SaeStoreDay (manages state for a store/transaction date). Tracks number of files exported and whether a store/day has been closed.
SAE_STORE_TRANSACTION (STAGING TABLE)	Persistence store for order store/day transaction header (units-of-work). Transactions staged for resa export.
SAE_STORE_TRANSACTION_DETAIL (STAGING TABLE)	Persistence store for order store/day transaction records. Detailed record data staged for resa export.
SAE_STORE_TRANSACTION_COUNTER	Persistence store for store next transaction number.

RCOM sales audit export triggering

During the customer order lifecycle, certain significant financial events (for example, a shipment, physical tender approval, return received, and so on) trigger objects within a customer order to be flagged as ready for Sales Audit export.

The sales audit export trigger is a code identifying whether a business object is ready for export or has been exported by the Sales Audit batch process.

- 'N' - not ready (or not applicable)
- 'R' - ready for sales audit processing
- 'E' - exported by resa (set after sales audit processes txn)

The following is a summary of the triggering events and the tables affected

PAID IN (for approval of physical tender)

cor_payment#sae_paid_state

cor_payment#sae_transaction_date (effective date of paid in/out txn, set when triggered)

PAID OUT (for refund payments for physcial tender, other special situations)

cor_payment#sae_paid_state

cor_payment#sae_transaction_date (effective date of paid in/out txn)

(for exchange sale cancellation merch reversal)

cor_order_transaction_group#merch_tender_liab_reversal_amt (amount for merch reversal)

cor_order_transaction_group#merch_tender_liab_sae_date (effective date of reversal)

PAID OUT (for post-sale accommodations)

cor_accommodation#sae_export_state

cor_accommodation#sae_transaction_date (effective date of sale txn, set when triggered)

cor_payment#sae_export_amount (updated with amount of ttend exported, always positive)

PAID OUT (Stored Value Cashout)

cst_stored_value_cashout (entire table used for triggering/processing txn)

RETURN (for return line being returned)

cor_return_line#sae_export_state

cor_return_line#sae_paid_in_exported_flag (flag indicating if return has been exported as paid-in transaction, used as trigger for paid-in reversal for cancelled replacement sale after replacement has been previously exported).

cor_return_line#sae_transaction_date (effective date of return txn, set when triggered)

cor_payment#sae_export_amount (updated with amount of ttend exported, always positive)

cor_transaction_group#sae_sale_export_amount (updated with amount of exchange sale merch ttend exported)

SALE (for normal shipment/sale)

cor_ship_container#sae_export_state

cor_ship_container#shipped_date (effective date of sale txn, set when triggered)

cor_payment#sae_export_amount (updated with amount of ttend exported, always positive)

cor_transaction_group#sae_return_export_amount (updated with amount of exchange return merch ttend exported)

cor_customer_order#outstanding_sae_goodwill_amt (updated during export if order is short within tolerance)

Additional detail about ResaRtlogTransformerBatch

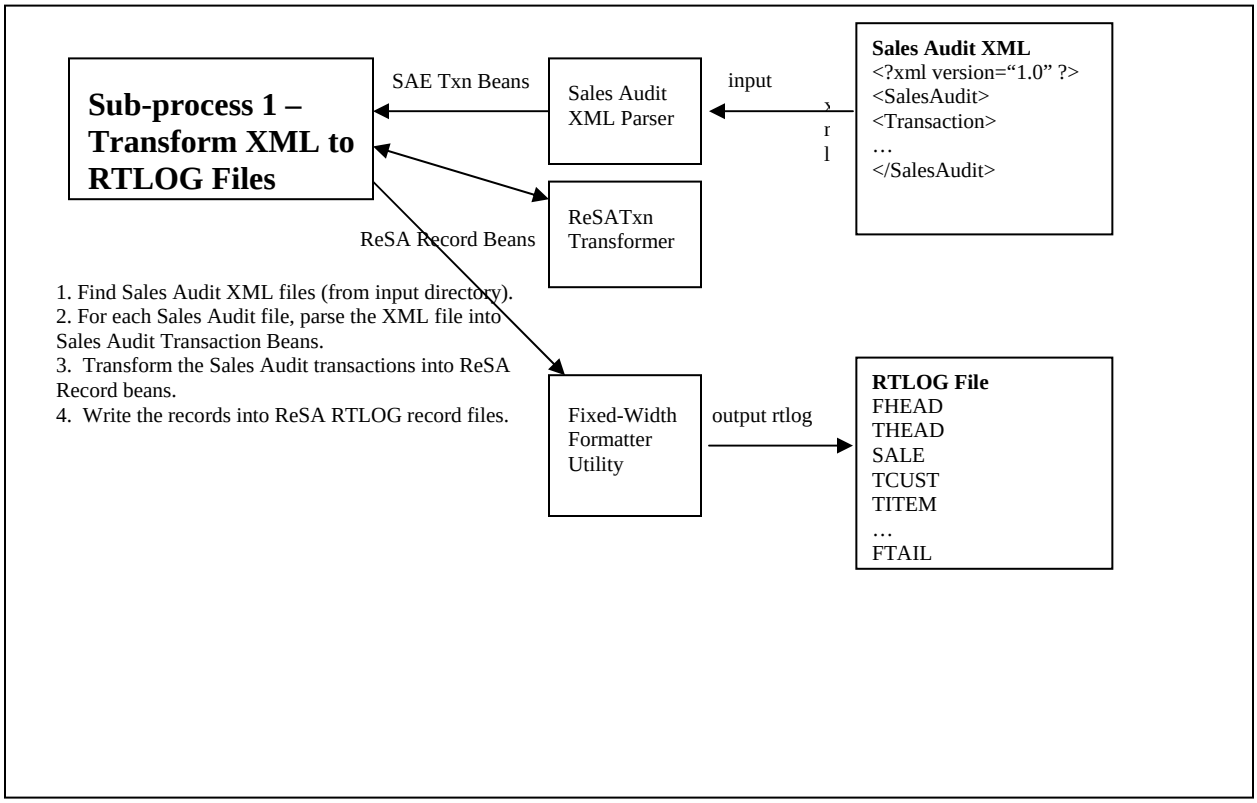
The following is a high-level overview of the processing steps within the ReSA RTLOG export batch process.

- Process Sales Audit XML files.
  This sub-process finds all Sales Audit XML export files from a specified input directory. Each file is parsed into one to many Sales Audit transaction beans (for a given store/day). The beans are sent through a conversion (translator) process to transform them into ReSA Record beans. The record beans are then sent through the Fixed Width Format utility to create an RTLOG flat file.

ReSA RTLOG transformer batch process flow

The following diagram depicts the basic process flow for the batch process supporting the ReSA RTLOG transformer batch program. This batch program is responsible for converting a Sales Audit XML export file into RTLOG file.

The transformer works from a specified input directory of sales audit xml files. For each file in the input directory, the batch program transforms it into an RTLOG file and writes the respective file to an **`../rtlog`** sub-directory. Upon successful transformation, the processed input xml file is moved from the input directory into the **`../processed`** sub-directory.



A note about multi-threading for CustomerFileExportBatch and CustomerFileImportBatch

Both CustomerFileExportBatch and CustomerFileImportBatch are designed to leverage Java threads to maximize throughpout for the jobs. The following batch arguments can be used to control the work settings.

- `-workerThreads`
  Controls the number of threads for the batch program. The default is 3 worker threads.
- `-workUnitSize`
  Controls the number of customers processed within a thread work unit. The default and recommended setting is 50.

For example:

```
-workerThreads 3 -workUnitSize 50
```

The two parts of the payment settlement batch process (for credit card payments only)

For credit card settlement records, the PaymentSettlementBatch process has been designed to include the two parts below.

1 The first part of the payment settlement batch is to process orders that have credit card payments triggered for settlement. The batch process then stages those settlement records into a staging table. An order is a unit of work.

2 For every settlement file to be produced, the batch process pulls records from the staging table and writes those records to a credit card settlement file. The batch process then purges those records from the staging table. A settlement file is a unit of work.

The three tables that are utilized in this batch processing are the following:

- PAY_SETTLEMENT_FILE (Credit card settlement file table)
- PAY_SETTLEMENT_RECORD (Credit card settlement unit-of-work header staging table)
- PAY_SETTLEMENT_RECORD_DETAIL (Credit card settlement transaction record staging table)

Help options for batch programs

Batch scripts provide `-help` options for details on any batch program.

The batch process, CustomerFileExportBatch, includes a complicated argument, `-searchCriteria`, that is explained below:

Required argument (for customer export)

`-searchCriteria` specification controls which customers are to be exported. This argument contains a ';' delimitted set of query parameters. The following are the supported criteria parts:

- active - flag indicating whether to export only active customers
- createdOrUpdatedAfterDate - date specifiying customers to export based on last updated or created, date format YYYYMMDDHHMMSS
- lastNameAlphaRange - alpha range of customers to export based on last name alpha prefix
- primaryBillToStates - criteria specifying primary bill-to state for customers to export

For example:

```
-searchCriteria active=true;lastNameAlphaRange=A~F;createdOrUpdatedAfterDate=20041001
-searchCriteria active=true;lastNameAlphaRange=A~F;primaryBillToStates=MN,WI
```

Return value batch standards

All batch processes in RCOM conform to the Retek batch standards. They are executed and terminated in the same manner as other batch processes in the Retek suite of products. The following guidelines describe the function return values and the program return values that RCOM's batch processes utilize:

Function Return Values

- **0** - The function completed without error, and processing should continue normally.
- **-1** - A fatal error occurred, and the calling function should also return -1, as should its calling function, and so on up to main(), where the final error messages are logged and the program is halted.
- **1** - A non-fatal error occurred (such as validation of an input record failed), and the calling function should either pass this error up another level or handle the exception.

Batch logging

Relevant progress messages are logged with regard to batch program runtime information. The setting for these log messages is at the Info level in log4j.

For more information, see the section, 'Logging standards' in "Chapter 2 – Backend system administration and configuration".

Chapter 4 – The RCOM export to a sales audit system (such as ReSA)

A sales audit system such as (Retek Sales Audit [ReSA]) is a tool that monitors the reliability and accuracy of transaction data gathered in other components of the enterprise, and compares the data to the rules and guidelines that a client establishes.

A functional overview of pre-sales audit processing

Before sales and other transactions data are sent to a sales audit system, RCOM typically processes data in the following way:

1 The warehouse sends RCOM the information that a container has been sent with items. As in the case of a standard sale transaction, for exchange sale items, the warehouse sends RCOM the information that a container has been sent with items. For returns and exchange returns, the warehouse sends RCOM the information that the item(s) have been returned. Once the item(s) is marked as returned, the transaction is sent to a sales audit system.

2 RCOM retrieves the order for its payment information. RCOM follows a settlement order (determining which payments should come first, and so on).

3 RCOM determines how much money applied to the container and takes that out of the payments. For example, if there were ten containers, the payment would be broken out in ten different ways. RCOM sends the final data to a sales audit system.

ReSA overview

ReSA provides an integrated flow of data from selling locations (POS and RCOM) to RMS, Retek Data Warehouse (RDW), and the general ledger. ReSA can accept transaction data from various front-end systems and move the data through a series of processes that culminates in 'clean data'. It flags inaccurate data for sales auditors, who can then correct the errors.

By running transactions from both the customer order management and point of sale applications through ReSA, a standard transaction data flow is enforced cross the entire enterprise. ReSA has the flexibility to define separate rules for the data flowing from the front-end system. Customer orders can thus be created on a transaction-by-transaction basis, and store sales can be processed on register-by-register or store basis.

ReSA provides an integrated flow of data from selling locations (POS and RCOM) to RMS, Retek Data Warehouse (RDW), and the general ledger. ReSA can accept transaction data from various front-end systems and move the data through a series of processes that culminates in 'clean data'. It flags inaccurate data for sales auditors, who can then correct the errors.

By running transactions from both the customer order management and point of sale applications through ReSA, a standard transaction data flow is enforced cross the entire enterprise.

Store day and the DCLOSE transaction type

'Store day' describes all transactions that occur in one RCOM business day. Because clients need the ability to audit transactions on a store-by-store basis for a defined period of time, store day data are maintained separately beginning with the initial import of data from the RCOM system.

Because RCOM has been designed to operate in a 24 hours and 7 day a week mode, ReSA includes functionality with regard to the DCLOSE transaction type. If the client is sending more than one file (as in, for example, a trickle polling situation), the client can specify the number of files that the system should expect in combination with the DCLOSE transaction type. This enhancement ensures that the system receives all of the files, even if the DCLOSE transaction type is, for some reason, received before the final file.

For example, if 24 files are expected over a given amount of time, and the file with the DCLOSE transaction type is, for some reason, sent before the 24th file, the RMS system will wait until the last file arrives before marking the store day record as partially or fully loaded in the database.

Store day is determined by the date of the returns and ship confirmations because that is primarily what RCOM sends through ReSA. RCOM looks at all of the containers that were ship confirmed since the previous ReSA export and then orders them by their confirmation date. At that point, if the date has switched over, from the 23rd to the 24th for example, the DCLOSE must be sent for the 23rd.

Multiple order lines and multiple stores

Ship containers can contain multiple order lines, which implies the use of multiple stores. RCOM processing looks at the ship container lines to determine which store the sale should be sent to. The same logic applies to shipping and handling. If a ship container is associated to two stores, the shipping and handling has to be broken out for each store.

Note that if a payment goes across two or more stores, the payment is divided to cover the cost of the item within the sales audit transaction.

Settlement order

Through the use of transaction tender (TTEND) records, RCOM provides the sales audit system with the correct settlement order (for example, cash before check, check before gift certificate, and so on). RCOM tracks the amount that has been sent with what payment method and then goes to the next payment method as necessary.

For example, if cash or check does not cover the container, RCOM can inform ReSA that a given part of the order is paid for by a gift certificate with this number, and so on.

RCOM does not send 'cancelled' credit cards data to the sales audit system.

Shipping and handling

Because of layout restrictions within the RTLOG, shipping and handling is sent from RCOM to ReSA as an item record (TITEM). An overview of the process follows:

1 Shipping and handling is set up as a nonmerchandise, noninventory, nonsellable item in the merchandising system (RMS).

2 The item is sent across the RIB for RCOM's subscription.

3 The item becomes a banner parameter in RCOM.

4 The parameter ID is placed as a TITEM record for shipping and handling in the RTLOG and sent to ReSA.

Sales audit dependencies on the merchandising system

ReSA depends upon the following data that is imported from the merchandising system.

- SKUs
  The following merchandising system-generated item number SKU values are sent to ReSA:
  - Shipping and handling
  - Personalization
  - Monogram
  - Gift card
  - Gift wrap

These items are set up at the banner level. There must be actual merchandising system generated SKUs set up for each banner. Each SKU must have item locations. The default value for each of the item number parameters is '9999999999'.

- The sales audit return reason codes import
  The sales audit return reason codes originate in tables within the merchandising system (such as RMS). These return reason codes must be uniform throughout the enterprise. Thus, RCOM subscribes to the code data over the RIB.
- Tender types (payment methods) and tender type IDs
  This seed data is populated through installation scripts. All codes and descriptions must match exactly across the enterprise. Otherwise, payments do not function properly in RCOM.

A note about tender types

Tender types can only be added to the system through the modification of 'hard' code. That is, tender types are not configurable.

The RTLOG overview

The Retek TLOG (RTLOG) is the sales download file to ReSA. The contents of each Retek TLOG file can be populated either per store per day or trickled throughout the day. RCOM is responsible for converting its transaction logs to RTLOGs. A file is created for each store-day combination. An RCOM batch process builds the RTLOG files that contain the sales transaction records.

Item line level media code

The RTLOG file that imports transactions from RCOM and the point of sale (POS) system has been updated to allow the media code to be communicated to RDW and RMS at the order header and at the item line level.

RCOM-related ReSA codes and types

The tables below illustrate codes and types specific to RCOM functionality. Existing base RMS codes and tenders are not included in the tables below.

Sub-transaction Types		
Description	**Code**	**Comments**
Return - Disposed	RETD	Sent on a RETURN transaction when the item disposition is disposed. No sub transaction type sent on normal return transaction.
Exchange In	EXCHI	Sent on a RETURN transaction when the return is associated with an exchange sale and not disposed.
Exchange In - Disposed	DEXCHI	Sent on a RETURN transaction when the return is associated with an exchange sale and has a disposition of disposed.
Exchange Out	EXCHO	Sent on a SALE transaction when the item is an exchange sale item.
Canceled exchange out item	EXCH	Sent on a PAIDOU transaction when the exchange sale item is cancelled.

Reason Codes		
Description	**Code**	**Comments**
Replacement In	1	When the replacement return item is required to be returned, a PAIDIN transaction is sent to increase the merchandise liability. This reason code is also used in COGS Adj records.
Replacement Out	2	When the replacement return item is required to be returned and the replacement sale is shipped, a PAIDOU is sent to decrease the merch liability for the item. This reason code is also used in COGS Adj records.
General Accommodation	CSTGEN	Credit accommodation created at the order header level for any other value besides shipping and handling, VAS, and taxes.
Tax Accommodation	CSTTAX	Credit accommodation created at the order header level for taxes. Sent as a PAIDOU transaction.
Payment to goodwill account	PTGW	When customer pays the underpayment amount after anything on the order is shipped, partially shipped, or cancelled, and the goodwill payment was not yet sent to RESA, then a PAIDIN is sent to RESA. The PAIDIN will have a reason code of 'PTGW'.
Customer overpayment refund	OVPY	When a refund is created because of a physical tender overpayment by the customer, OR the customer paid with physical tender for an exchange sale that is now cancelled, the PAIDOU has this reason code.
Customer refund for cancelled exchange sales	CSTRFD	When an exchange sale is cancelled and RCOM generates a refund, a PAIDOU is sent to ReSA with this reason code.
Back out merch liability for cancelled exchange sales	CANCEL	When an exchange sale is cancelled, RCOM sends a PAIDOU to ReSA with this reason code to decrease the merch liability that was increased with the exchange return.

Discount types		
Description	**Code**	**Comments**
Shipping and Handling Accommodation	CSTSH	Sent on the IDISC record for order line accommodations when the accommodation is for shipping and handling.
Value Added Service Accommodation	CSTVAS	Sent on the IDISC record for order line accommodations when the accommodation is for VAS.
Merchandise Accommodation	CSTMER	Sent on the IDISC record for order line accommodations when the accommodation is for merchandise value.
General RCOM Discount	G	Sent on the IDISC record for general discount promotions in RCOM.

Inv Adj Reasons		
Description	**Code**	**Comments**
Customer Return	60	Return disposition is not mapped to a disposed transaction – both return and exchange return items
Disposed Customer Return	61	Return disposition is mapped to a disposed transaction – both return and exchange return items
No Item Customer Return	62	Return is not required – both return and exchange return items
Replacement In	65	When replacement return confirmation says the replacement return item is NOT disposed.

Tender Type Groups		
Description	**Code**	**Comments**
Contra-Sales/Liability	CSLI	Tender group associated to liability for exchanges, replacements, and refunds.
Voucher Redemption	VOUCHR	Tender group sent to RESA for a SALE transaction with a voucher. This is a placeholder TTEND value, so the voucher is not redeemed twice in RESA.

Tender Type Groups		
Description	**Code**	**Comments**
Customer Goodwill	GOODW	Tender group sent to RESA when an order has an outstanding balance due to W-S, but is within the underpayment tolerance. Sent on the SALE transaction.
Voucher	VOUCH	Tender group sent to RESA for a PAIDIN transaction. This redeems the voucher.

Tender Type IDs		
Description	**Code**	**Comments**
Customer Liability – Check (tender type group = CSLI)	10100	Used on sales transactions to reverse liability from initial paid in transaction.
Customer Liability – Cash (tender type group = CSLI)	10200	Used on sales transactions to reverse liability from initial paid in transaction.
Merchandise Liability (tender type group = CSLI)	10400	Tender ID used for merch liability of exchange return and sales.
Refund Liability - Merch Certificate (tender type group = CSLI)	10500	Used for merch voucher refund to customer. This should account for the lag in time between receiving the payment and sending out the refund. Retek will not systematically relieve these liability accounts.
Refund Liability – Check (tender type group = CSLI)	10600	Used for check refund to customer. This should account for the lag in time between receiving the payment and sending out the refund. Retek will not systematically relieve these liability accounts.
Replacements (tender type group = CSLI)	10700	When the replacement return item is required to be returned, the merch liability is increased using this tender ID. When the replacement sale item is shipped, this tender ID decreases merch liability.
Reward Certificate Liability (tender type group = CSLI)	10300	Used on sales transactions to reverse liability from initial paid in transaction.
Gift Certificate Redemption (tender type group = VOUCHR)	4500	Tender type ID sent to RESA for a SALE transaction with a gift certificate. This is a placeholder TTEND value, so the gift certificate is not redeemed twice in RESA.

Tender Type IDs		
Description	**Code**	**Comments**
Merch Voucher Redemption (tender type group = VOUCHR)	4510	Tender type ID sent to ReSA for a SALE transaction with a merch voucher. This is a placeholder TTEND value, so the merch voucher is not redeemed twice in RESA.
Reward Certificate (tender type group = VOUCHR)	4050	Tender type ID sent to ReSA for PAIDIN transaction with reward certificate. This will redeem the reward certificate.
Customer Goodwill (tender type group = GOODW)	11000	Tender ID sent to ReSA when an order has an outstanding balance due to W-S, but is within the underpayment tolerance. Sent on the SALE transaction.
Gift Card (tender type group = VOUCHR)	4060	Tender type ID sent to ReSA for PAIDIN transaction with gift card. This will redeem the certificate.
Gift Card Redemption (tender type group = VOUCHR)	4520	Tender ID sent to ReSA for a SALE transaction with a gift card. This is a placeholder TTEND value, so the gift card is not redeemed twice in ReSA.
Merchandise Card (tender type group = VOUCHR)	4070	Tender type ID sent to ReSA for PAIDIN transaction with a merchandise card. This will redeem the certificate.
Merchandise Card Redemption (tender type group = VOUCHR)	4530	Tender ID sent to ReSA for a SALE transaction with a merchandise card. This is a placeholder TTEND value, so the gift card is not redeemed twice in ReSA.

Promotion Numbers		
Description	**Code**	**Comments**
DTC Promotion	2000	Sent on the IDISC record for general discount promotions in RCOM.
In store discount	1004	Sent on the IDISC record for order line accommodations.

RCOM-specific business rules for the RTLOG

- The filename convention is RTLOG_STORE_BUSINESSDATE_CREATEDATETIME.DAT (for example, RTLOG_0001_12102002121020021603 00.DAT). The BUSINESSDATE is the shipped date of the orders. The CREATEDATETIME is populated with the DATETIME the file creation process was started.
- A ship container is associated to a ship request, which is associated to a customer order and customer.
- A ship container is associated to a ship request; take the ship_to_address_id and customer_id from ship_request and look at the customer_address table to get the customer name. The address information is from the address table.
- DCLOSE is determined by looking for a change in transaction date and no transactions with exceptions for that day/store. An exception causes an order to not get sent to RESA. For example, if there are transactions for the 4th, 4th, 4th, and then the 5th, and the transactions for the 4th do not have an exception, a DCLOSE record should be created for the 4th; a new file is created for the 5th. If there is an exception on the 4th, the DCLOSE is not sent, and a manual investigation will need to figure out the issue. The system does not send the DCLOSE for the day until all exceptions for the store/day are handled.
- Because disposition codes are sent to RESA, each RETURN transaction has only 1 ‘regular’ TITEM element (the S&H TITEM and VAS TITEM are still included).
- A tender type is sent to RESA for exchange and replacement items, as well as for customer liability for refunds. The tender type group is CSLI - Contra-Sales/Liability; its tender type id will be 10400.
- There is a tender type sent to RESA for vouchers, VOUCHR. It is sent for a SALE transaction with a voucher. This is a placeholder TTEND value, so the voucher is not redeemed twice in RESA. VOUCH is sent to RESA on a PAIDIN transaction to redeem the voucher.
- A tender type for RESA’s use covers customer underpayment. It is a ‘goodwill’ tender of type GOODW and ID 11000. The underpayment value is a TTEND in the transaction to RESA and is sent to RESA during the SALE transaction.
- When the customer pays the underpayment amount after anything of the order is shipped, partially shipped, or cancelled, and the goodwill payment has not yet been sent to RESA, then a PAIDIN is sent to RESA. The PAIDIN contains a THEAD reason code of ‘PTGW’. This applies to any tender type used to cover the underpayment amount.
- For accommodations applied after the order has a shipped qty, a zero order/zero unit sale transaction is created. The TTEND = SUM(TITEM) + TTAX – SUM(IDISC). TITEM has a 0 unit_retail and 0 original_unit_retail price and a 0 qty. The transaction type is SALE.

Appendix A – Batch file layout specifications

WebMedia data export schema used in internet batch processing

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

<xsd:element name="WebMedia" type="WebMediaType"/>

<xsd:simpleType name="positiveMoney">
  <xsd:restriction base="xsd:decimal">
    <xsd:fractionDigits value="2"/>
<!--        <xsd:minInclusive value="0"/>< castor bug, fixed in 0.9.5.2-->
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="positiveQuantity">
  <xsd:restriction base="xsd:decimal">
    <xsd:fractionDigits value="4"/>
<!--    <xsd:minInclusive value="0"/> castor bug, fixed in 0.9.5.2-->
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="positivePercent">
  <xsd:restriction base="xsd:decimal">
<!--    <xsd:minInclusive value="0"/> castor bug, fixed in 0.9.5.2-->
    <xsd:maxExclusive value="100"/>
  </xsd:restriction>
</xsd:simpleType>

 <xsd:complexType name="WebMediaType">
  <xsd:sequence>
   <xsd:element name="Description" type="xsd:string"/>
   <xsd:element name="BannerName" type="xsd:string"/>
   <xsd:element name="Season" type="xsd:string"  minOccurs="0"/>
   <xsd:element name="ActiveDate"  type="xsd:date"  minOccurs="0"/>
<!--   <xsd:element name="InactiveDate"  type="xsd:date"  minOccurs="0" />
Media has no inactive date -->
```

```
   <xsd:element name="WebShippingRateTable" type="WebShippingRateTableType"
/>
   <xsd:element name="WebGiftServiceList" type="WebGiftServiceListType"
minOccurs="0" />
   <xsd:element name="WebSellingItemList" type="WebSellingItemListType" />
   <xsd:element name="WebTenderTypeList" type="WebTenderTypeListType" />
  </xsd:sequence>
 </xsd:complexType>


 <xsd:complexType name="WebShippingRateTableType">
  <xsd:sequence>
   <xsd:element name="RushCharge" type="positiveMoney" minOccurs="0"/>
   <xsd:element name="ExceptionalRushCharge" type="positiveMoney"
minOccurs="0"/>
   <xsd:element name="WebShippingRate" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType> <!-- Range -->
     <xsd:sequence>
      <xsd:element name="RangeMin" type="positiveMoney"/>
      <xsd:element name="RangeMax" type="positiveMoney"/>
      <xsd:choice>
       <xsd:element name="FlatRate" type="positiveMoney"/>
       <xsd:element name="PercentRate" type="positivePercent"/>
      </xsd:choice>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>


 <xsd:complexType name="WebGiftServiceListType">
  <xsd:sequence>
   <xsd:element name="WebGiftService" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="Type" type="xsd:string"/>
      <xsd:element name="Description" type="xsd:string"/>
      <xsd:element name="Price" type="positiveMoney"/>
      <xsd:element name="StartDate" type="xsd:date"/>
      <xsd:element name="EndDate" type="xsd:date" minOccurs="0"/>
     </xsd:sequence>
    </xsd:complexType>
```

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


 <xsd:complexType name="WebSellingItemListType">
  <xsd:sequence>
   <xsd:element name="WebSellingItem" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="Description" type="xsd:string"/>
      <xsd:element name="DepartmentId" type="merchHierarchyId"
minOccurs="0"/>
      <xsd:element name="ClassId" type="merchHierarchyId" minOccurs="0"/>
      <xsd:element name="SubclassId" type="merchHierarchyId" minOccurs="0"/>
      <xsd:element name="SpecialShippingDescription" type="xsd:string"
minOccurs="0"/>
<!--      <xsd:element name="perishableFlag" type="xsd:boolean"/>  -->
      <xsd:element name="WebUpSellList" type="WebAlternateSellingListType"
minOccurs="0"/>
      <xsd:element name="WebCrossSellList"
type="WebAlternateSellingListType" minOccurs="0"/>
<!--  <xsd:element name="HandlingTemperatureCode" type="xsd:string"/>
      <xsd:element name="HandlingSensitivityCode" type="xsd:string"/>
      <xsd:element name="merchandiseFlag" type="xsd:string"/>
      <xsd:element name="ShipAloneFlag" type="xsd:boolean"/>
      <xsd:element name="taxable" type="xsd:boolean"/>
-->
      <xsd:choice>
       <xsd:element name="WebSellingItemList"
type="WebSellingItemListType"/> <!-- Multi-Style Selling Item -->
       <xsd:element name="WebSellingSkuList" type="WebSellingSkuListType"/>
      </xsd:choice>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>


 <xsd:complexType name="WebSellingSkuListType">
  <xsd:sequence>
   <xsd:element name="WebSellingSku" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
```

```
      <xsd:sequence>
       <xsd:element name="WebSellingSkuId" type="WebSellingSkuIdType"/>
       <xsd:element name="Description" type="xsd:string" minOccurs="0"/>
       <xsd:element name="WebDifferentiatorList"
type="WebDifferentiatorListType" minOccurs="0"/>
       <xsd:element name="AdditionalDomesticShippingFlatRate"
type="positiveMoney" minOccurs="0"/>
       <xsd:element name="AdditionalInternationalShippingFlatRate"
type="positiveMoney" minOccurs="0"/>
       <xsd:element name="RegularPrice" type="positiveMoney"/>
<!--      <xsd:element name="SalePrice" type="positiveMoney"
nillable="true"/> how do we calculate sale price -->
       <xsd:element name="IsGiftWrappable" type="xsd:boolean"/>
       <xsd:element name="IsDirectShip" type="xsd:boolean"/>
       <xsd:element name="UnitOfMeasureDescription" type="xsd:string"/>
       <xsd:choice>
        <xsd:element name="WebSku">
         <xsd:complexType>
           <xsd:sequence>
            <xsd:element name="id" type="xsd:string"/>
           <xsd:element name="WebValueAddedServiceList"
type="WebValueAddedServiceListType" minOccurs="0"/>
           </xsd:sequence>
         </xsd:complexType>
       </xsd:element>
        <xsd:element name="WebPack" type="WebPackType"/>
       </xsd:choice>
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
   </xsd:sequence>
  </xsd:complexType>

<xsd:simpleType name="WebSellingSkuIdType">
   <xsd:restriction base="xsd:string">
    <xsd:pattern value="[0-9]{2,3}-[0-9]{1,12}"/>
   </xsd:restriction>
  </xsd:simpleType>

<xsd:simpleType name="merchHierarchyId">
   <xsd:restriction base="xsd:string">
    <xsd:maxLength value="4"/>
```

```
  </xsd:restriction>
 </xsd:simpleType>

 <xsd:complexType name="WebAlternateSellingListType">
  <xsd:sequence>
   <xsd:element name="WebSellingSkuId" type="WebSellingSkuIdType"
minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="WebFontListType">
  <xsd:sequence>
   <xsd:element name="Font" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="FontId" type="xsd:string"/>
       <xsd:element name="FontDescription" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="WebColorListType">
  <xsd:sequence>
   <xsd:element name="Color" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="ColorId" type="xsd:string"/>
       <xsd:element name="ColorDescription" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="WebPersonalizationLineListType">
  <xsd:sequence>
   <xsd:element name="WebPersonalizationLine" minOccurs="0"
maxOccurs="unbounded">
    <xsd:complexType>
```

```
      <xsd:sequence>
       <xsd:element name="LineNumber" type="xsd:positiveInteger"/>
       <xsd:element name="Enabled" type="xsd:boolean"/>
       <xsd:element name="MaxCharacters" type="xsd:positiveInteger"/>
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
   </xsd:sequence>
  </xsd:complexType>


  <xsd:complexType name="WebValueAddedServiceListType">
   <xsd:sequence>
    <xsd:element name="WebValueAddedService" minOccurs="0"
maxOccurs="unbounded">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="WebVasType" minOccurs="1">
        <xsd:simpleType>
         <xsd:restriction base="xsd:NMTOKEN">
          <xsd:enumeration value="Personalization"/>
          <xsd:enumeration value="Monogramming"/>
         </xsd:restriction>
        </xsd:simpleType>
       </xsd:element>
       <xsd:element name="TypeCode" type="xsd:string"/>
       <xsd:element name="TypeDescription" type="xsd:string"/>
       <xsd:element name="Supplier" type="xsd:string"/>
       <xsd:element name="UnitPrice" type="positiveMoney"/>
       <xsd:element name="WebFontList" type="WebFontListType" minOccurs="0"/>
       <xsd:element name="WebColorList" type="WebColorListType"
minOccurs="0"/>
       <xsd:choice>
        <xsd:element name="WebPersonalizationLineList"
type="WebPersonalizationLineListType" nillable="true"/>
        <xsd:element name="WebMonogramInformation">
         <xsd:complexType>
          <xsd:sequence>
           <xsd:element name="CharacterMonogramingOneFlag"
type="xsd:boolean"/>
           <xsd:element name="CharacterMonogramingTwoFlag"
type="xsd:boolean"/>
```

```
          <xsd:element name="CharacterMonogramingThreeFlag"
type="xsd:boolean"/>
         </xsd:sequence>
        </xsd:complexType>
       </xsd:element>
       </xsd:choice>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="WebDifferentiatorListType">
  <xsd:sequence>
   <xsd:element name="WebDifferentiator" minOccurs="1"
maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="DiffTypeDescription" type="xsd:string"/>
      <xsd:element name="DiffDescription" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="WebPackType">
  <xsd:sequence>
   <xsd:element name="PackId" type="xsd:string"/>
   <xsd:element name="WebPackComponentList"
type="WebPackComponentListType"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="WebPackComponentListType">
   <xsd:sequence>
    <xsd:element name="WebPackComponent" minOccurs="1"
maxOccurs="unbounded">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:choice>
```

```
          <xsd:element name="WebSku" type="WebSkuType"/>
          <xsd:element name="WebPack" type="WebPackType"/>
         </xsd:choice>
         <xsd:element name="Quantity" type="positiveQuantity"/>
        </xsd:sequence>
       </xsd:complexType>
      </xsd:element>
     </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="WebTenderTypeListType">
     <xsd:sequence>
      <xsd:element name="WebTenderType" minOccurs="1" maxOccurs="unbounded">
       <xsd:complexType>
        <xsd:sequence>
         <xsd:element name="Id" type="xsd:string" minOccurs="1"/>
         <xsd:element name="groupId" type="xsd:string" minOccurs="1"/>
         <xsd:element name="Description" type="xsd:string"/>
        </xsd:sequence>
       </xsd:complexType>
      </xsd:element>
     </xsd:sequence>
    </xsd:complexType>

   </xsd:schema>
```

XML file layout – customer import/export integration

<CUSTOMER_FILE> Element

All XML-based data formats require a 'root element'. This element provides parsing applications with a definitive place to start reading the encapsulated data. The root is element is < CUSTOMER_MERGE >.

There is no current support included for the following typical root element attributes:

- XML namespace
- Version
- Name

There are one to many <CUSTOMER> elements that can appear within the <CUSTOMER_FILE> root element.

<CUSTOMER> Element

The <CUSTOMER> element can only appear under the root element for each request within the document. The <CUSTOMER> has two attributes: ID, and type.

Attribute Name	Data Type/Value	Description	Required
id	String	The customer identifier that the customer order is associated to.	Yes
type	Import	Customer import request	Yes
	Export	Customer export request	

Element Name	Data Type/Value	Description	Required
action	A, U	The action for this import request.	Yes (for Import only)
householdNumber	String	Primary household number for the customer.	No
subAccountNumber	String	Sub-account number for the customer.	No

Element Name	Data Type/Value	Description	Required
acquisitionMethodCode	B,S,C,I,G	Code identifying the acquisition method for this customer. B – Buyer S – Ship To Customer C – Catalog request I – Internet G – Gift Registry	Yes
initialBannerCode	String	Default banner code for this customer.	No
initialSourceCode	String	Default media source code for this customer.	No
originalOrderDate	YYYYMMDDHHMMSS	Date of the first order taken for this customer.	No
activeFlag	Y/N	Flag indicating if this customer is active.	Yes
inactiveReasonCode	String	Code identifying the reason that the customer is inactive (if active flag is Y).	No
createdByUser	String	User name of the created by user.	Yes – for a create No – for an update
createDate	YYYYMMDDHHMMSS	Create date for the address.	Yes – for a create No – for an update
CUSTOMER_ADDRESS*	N/A	Address element	Yes
CUSTOMER_TELEPHONE*	N/A	Telephone element	Yes
CUSTOMER_EMAIL*	N/A	Email element	Yes
CUSTOMER_ALTERNATE_NUMBER*	N/A	Alternate number element	No

<CUSTOMER_ADDRESS> Element

Attribute Name	Data Type/Value	Description	Required
id	String	The unique identifier for this customer address.	Yes

Element Name	Data Type/Value	Description	Required
nameTitleCode	String	The name title associated to the customer for this address (ex. Dr, Mr, Mrs).	No
firstName	String	The first name associated to the customer for this address.	Yes
middleInitial	Char(1)	The middle initial associated to the customer for this address.	No
lastName	String	The last name associated to the customer for this address.	Yes
nameSuffixCode	String	The name suffix associated to the customer for this address (ex. Jr, Sr, III).	No
primaryBillToAddressFlag	Y/N	Flag indicating if this is the primary bill-to address for the customer.	Yes
primaryShipToAddressFlag	Y/N	Flag indicating if this is the primary ship-to address.	Yes
billToAddressFlag	Y/N	Flag indicating if this a bill-to address for the customer.	Yes
addressLabel	String	The label for this address.	Yes
dayTelephoneNumber	String	The day phone number associated to the customer address.	No
dayTelephoneExtension	String	The day phone number extension associated to the customer address.	No
eveningTelephoneNumber	String	The evening phone associated to the customer address.	No
eveningTelephoneExtension	String	The evening phone extension associated to the customer address.	No

Element Name	Data Type/Value	Description	Required
emailAddress	String	The primary email associated to the customer address.	No
addressLine1	String	The first line associated to the customer address.	Yes
addressLine2	String	The second line associated to the customer address.	No
addressLine3	String	The third line associated to the customer address.	No
attention	String	The attention line associate to the customer address.	No
city	String	The city associated to the customer address.	Yes
state	String	The state associated to the customer address.	Yes
postalCode	String	The postal code associated to the customer address.	Yes
countryCode	String	The country code associated to the customer address.	Yes
countyName	String	The county associated to the customer address.	No
taxReferenceCode	String	The tax reference code for the customer address.	No
changeReasonCode	String	Code identifying a change reason for this customer address (for its last update).	No
giftReceiptCustomerId	String	Gift recipient customer identifier.	No
activeFlag	Y/N	Flag indicating if this is an active address.	Yes
createdByUser	String	User name of the created by user.	Yes – for a create No – for an update
createDate	YYYYMMDDHHMMSS	Create date for the address.	Yes – for a create No – for an update
lastUpdatedByUser	String	User name of the last update user.	No

Element Name	Data Type/Value	Description	Required
lastUpdateDate	YYYYMMDDHHMMSS	Last update date of the address.	No

<CUSTOMER_TELEPHONE > Element

Attribute Name	Data Type/Value	Description	Required
id	String	The unique identifier of the customer telephone.	Yes

Element Name	Data Type/Value	Description	Required
telephoneNumber	String	The telehphone number.	No
Extension	String	The extension of the telephone number.	No
primaryDayTelephoneFlag	Y/N	Flag indicating if this is the primary day telephone number.	Yes
primaryEveningTelephoneFlag	Y/N	Flag indicating if this is the primary evening telephone number.	Yes
activeFlag	Y/N	Flag indicating if this is an active telephone number.	Yes
createdByUser	String	User name of the created by user.	Yes – for a create No – for an update
createDate	YYYYMMDDHHMMSS	Create date for the telephone.	Yes – for a create No – for an update
lastUpdatedByUser	String	User name of the last update user.	No
lastUpdateDate	YYYYMMDDHHMMSS	Last update date of the telephone.	No

<CUSTOMER_EMAIL > Element

Attribute Name	Data Type/Value	Description	Required
id	String	The customer identifier that the customer order is associated to.	Yes

Element Name	Data Type/Value	Description	Required
emailAddress	String	The primary email associated to the customer address.	Yes
primaryEmailFlag	Y/N	Flag indicating if this is the primary email address.	Yes
activeFlag	Y/N	Flag indicating if this is an active email address.	Yes
createdByUser	String	User name of the created by user.	Yes – for a create No – for an update
createDate	YYYYMMDDHHMMSS	Create date for the email.	Yes – for a create No – for an update
lastUpdatedByUser	String	User name of the last update user.	No
lastUpdateDate	YYYYMMDDHHMMSS	Last update date of the email.	No

<CUSTOMER_ALTERNATE_NUMBER > Element

Attribute Name	Data Type/Value	Description	Required
type	String H,A	The type of the alternate customer number. H – household number A – Alternate customer number	Yes

Element Name	Data Type/Value	Description	Required
alternateNumber	String	The primary email associated to the customer address.	Yes

<CUSTOMER_PREFERENCE > Element

Attribute Name	Data Type/Value	Description	Required
id	String	The unique identifier for the customer preference.	Yes

Element Name	Data Type/Value	Description	Required
bannerCode	String	The primary email associated to the customer address.	Yes
doNotShareAddressFlag	Y/N	Flag indicating if customer wishes not to share address.	Yes
doNotShareEmailFlag	Y/N	Flag indicating if customer wishes not to email address.	Yes
doNotMailFlag	Y/N	Flag indicating if customer wishes not to receive solicitation mail.	Yes
doNotCallFlag	Y/N	Flag indicating if customer wishes not to receive solicitation calls.	Yes
doNotEmailFlag	Y/N	Flag indicating if customer wishes to not to receive solication emails.	Yes
creditCardOptOutFlag	Y/N	Flag indicating if customer wishes ??.	Yes
contactMethodCode	String E,T,M,F	Code identifying customer's preferred contact method. E – Email T – Telephone M – Mail F - Fax	No
contactTimeTypeCode	String 1-7	Code identifying customer's preferred contact type. 1- Sunday … 2 - Monday 7 - Saturday	No

Element Name	Data Type/Value	Description	Required
createdByUser	String	User name of the created by user.	Yes – for a create No – for an update
createDate	YYYYMMDDHHMMSS	Create date for the email.	Yes – for a create No – for an update
lastUpdatedByUser	String	User name of the last update user.	No
lastUpdateDate	YYYYMMDDHHMMSS	Last update date of the email.	No

File layout definition for customer imports

```
<CustomerFile createTime="20041021174824">
   <Customer id="1040001098398951203" type="A">
<householdNumber>1234</householdNumber>
<subAccountNumber>12345</subAccountNumber>
<acquisitionMethodCode>B</acquisitionMethodCode>
<initialBannerCode>500</initialBannerCode>
<activeFlag>Y</activeFlag>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
   <CustomerAddress>
<nameTitleCode>MR</nameTitleCode>
<firstName>Alina</firstName>
<middleInitial>A</middleInitial>
<lastName>Begel</lastName>
<primaryBillToAddressFlag>Y</primaryBillToAddressFlag>
<primaryShipToAddressFlag>N</primaryShipToAddressFlag>
<billToAddressFlag>Y</billToAddressFlag>
<addressLabel>Home</addressLabel>
<dayTelephoneNumber>6125551234</dayTelephoneNumber>
<dayTelephoneExtension>9876</dayTelephoneExtension>
<eveningTelephoneNumber>6125554321</eveningTelephoneNumber>
<emailAddress>alina.begel@yahoo.com</emailAddress>
<addressLine1>1234 Happy Lane</addressLine1>
<addressLine2>Apt. 321</addressLine2>
<city>Minneapolis</city>
```

```
<state>MN</state>
<postalCode>55443</postalCode>
<countryCode>USA</countryCode>
<countyName>HENNEPIN</countyName>
<activeFlag>Y</activeFlag>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
</CustomerAddress>
   <CustomerAddress>
<nameTitleCode>MR</nameTitleCode>
<firstName>Alina</firstName>
<middleInitial>A</middleInitial>
<lastName>Begel</lastName>
<primaryBillToAddressFlag>N</primaryBillToAddressFlag>
<primaryShipToAddressFlag>Y</primaryShipToAddressFlag>
<billToAddressFlag>N</billToAddressFlag>
<addressLabel>Work</addressLabel>
<dayTelephoneNumber>6125551234</dayTelephoneNumber>
<dayTelephoneExtension>9876</dayTelephoneExtension>
<eveningTelephoneNumber>6125554321</eveningTelephoneNumber>
<emailAddress>alina.begel@yahoo.com</emailAddress>
<addressLine1>4321 Tcb Ave.</addressLine1>
<addressLine2>Suite 101</addressLine2>
<city>Minneapolis</city>
<state>MN</state>
<postalCode>55443</postalCode>
<countryCode>USA</countryCode>
<countyName>HENNEPIN</countyName>
<activeFlag>Y</activeFlag>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
</CustomerAddress>
   <CustomerTelephone>
<telephoneNumber>6125551234</telephoneNumber>
<extension>9876</extension>
<primaryDayTelephoneFlag>Y</primaryDayTelephoneFlag>
<primaryEveningTelephoneFlag>N</primaryEveningTelephoneFlag>
<activeFlag>Y</activeFlag>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
```

```
</CustomerTelephone>
   <CustomerTelephone>
<telephoneNumber>6125554321</telephoneNumber>
<primaryDayTelephoneFlag>N</primaryDayTelephoneFlag>
<primaryEveningTelephoneFlag>Y</primaryEveningTelephoneFlag>
<activeFlag>Y</activeFlag>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
</CustomerTelephone>
   <CustomerEmail>
<emailAddress>alina.begel@yahoo.com</emailAddress>
<activeFlag>Y</activeFlag>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
</CustomerEmail>
   <CustomerPreference>
<bannerCode>500</bannerCode>
<doNotShareAddressFlag>Y</doNotShareAddressFlag>
<doNotShareEmailFlag>Y</doNotShareEmailFlag>
<doNotMailFlag>Y</doNotMailFlag>
<doNotCallFlag>Y</doNotCallFlag>
<doNotEmailFlag>Y</doNotEmailFlag>
<creditCardOptOutFlag>N</creditCardOptOutFlag>
<mailOnlyOncePerSeasonFlag>Y</mailOnlyOncePerSeasonFlag>
<contactMethodCode>E</contactMethodCode>
<contactTimeTypeCode>E</contactTimeTypeCode>
<createdByUser>batch</createdByUser>
<createDate>20041021174911</createDate>
</CustomerPreference>
</Customer>
</CustomerFile>
```

XML file layout – customer merge integration

<CUSTOMER_MERGE> Element

All XML-based data formats require a 'root element'. This provides parsing applications with a definitive place to start reading the encapsulated data. The root is element is < CUSTOMER_MERGE >.

There is no current support included for the following typical root element attributes:

- XML namespace
- Version
- Name

There are one to many <CUSTOMER_MERGE_REQUEST> elements that can appear within the <CUSTOMER_MERGE> root element.

<CUSTOMER_MERGE_REQUEST> Element

The < CUSTOMER_MERGE_REQUEST > element can only appear under the root element for each request within the document. The < CUSTOMER_MERGE_REQUEST> has two attributes: ID, and type.

Attribute Name	Values	Description	Required
id	Number	Unique identifier for the customer merge request.	Yes
type	Import	Customer merge import request	Yes
	Export	Customer merge export request	

<CUSTOMER_MERGE_REQUEST type=Import>

Element Name	Data Type/Value	Description	Required
createDate	YYYYMMDDHHMMSS	Date of the merge request.	Yes
MERGE_CUSTOMER*	N/A	Customer merge request customer element.	Yes

<CUSTOMER_MERGE_REQUEST type=Export>

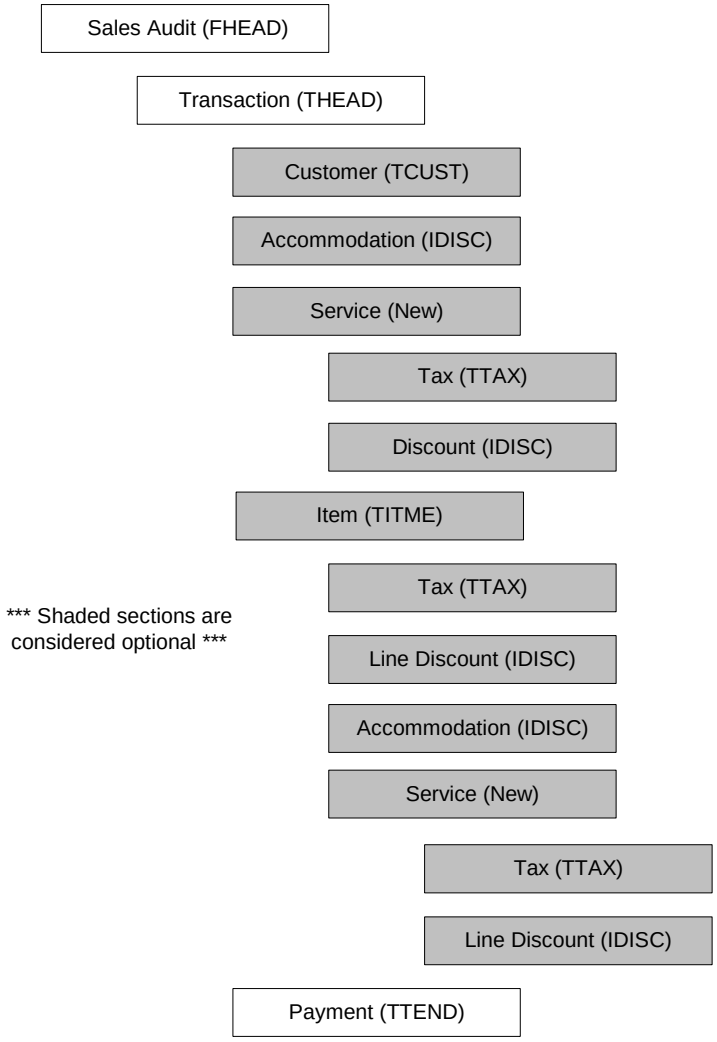
Element Name	Data Type/Value	Description	Required
createDate	YYYYMMDDHHMMSS	Date of the merge request.	Yes
createdByUser	String	Unique username of the user that created this request.	Yes
MERGE_CUSTOMER*	N/A	Customer merge request customer element.	Yes

<MERGE_CUSTOMER> Element

Attribute Name	Data Type/Value	Description	Required
id	String	The customer identifier that the customer order is associated to.	Yes

Element Name	Data Type/Value	Description	Required
mergeStatus	A – Active customer M – Merged customer	The merge status of the customer for this merge request.	Yes
householdNumber	Char(12)	The household number associated to this customer.	No
subAccountNumber	Char(12)	The sub-account number associated to this customer.	No

XML file layout for sales audit system export



<SALES_AUDIT> Element

All XML-based data formats require a 'root element'. This provides parsing applications with a definitive place to start reading the encapsulated data. The root is element is < SALES_AUDIT >.

There is no current support included for the following typical root element attributes:

- XML namespace
- Version
- Name

There are one to many <TRANSACTION> elements that can appear within the <SALES_AUDIT > root element.

<TRANSACTION> Element

The <TRANSACTION> element can only appear under the root element for each transaction within the document. The <TRANSACTION> has two attributes: tranNumber, ID, and type. This is known as the THEAD portion of the Retek TLOG (RTLOG).

Attribute Name	Values	Description	Required
Id	SSSS*NNNN	Sales Audit ID will be used to uniquely identify the transaction for the batch converter. RCOM specific.	Yes
Type	Sale	Sale transactions	Yes
	Return	Return transactions	
	PaidOut	Paid-Out transactions	
	PaidIn	Paid-In transactions	
	Dclose	Drawer Close transactions	

<TRANSACTION type=Sale>

Element Name	Data Type/Value	Description	Required
transactionDate	YYYYMMDDHHMMSS	Date of the transaction.	Yes
transactionNumber	Number	A unique number used to identify each transaction.	Yes
subTransactionType	EXCHO, CACCOM	EXCHO – Exchange out CACCOM – Customer Accomodation (Post Sale Merchandise)	No
storeNumber	String	The Store number that the transaction will be attributed to.	Yes
orderStoreNumber	String	Physical store entered within the RCOM application.	No
customerOrderNumber	String	The Customer Order that generated the transaction.	Yes

Element Name	Data Type/Value	Description	Required
customerOrderType	S, E	Order Type values include “S” for Standard and “E” for Employee	Yes
customerOrderSource	T, M, I, G	Order source of the current order. Current values included Telephone, Mail Order, Internet, Gift Registry	Yes
customerOrderDate	YYYYMMDDHHMMSS	Create date of the order.	Yes
bannerCode	String	The banner code that is associated to the customer order header.	Yes
mediaCode	String	The media code that is associated to the customer order header. This can be different than the media code at the line level.	Yes
orderAcceptanceUser	String	The user name of the user that initiated the sale.	Yes
employeeNumber	String	For RCOM XI, this value will reflect 1 for an employee sale (and mail order order type) and a –1 for a standard order.	No
CUSTOMER	N/A	Customer Transaction Element	Yes
SERVICE*	N/A	Service Transaction Element	No
ITEM+	N/A	Item Transaction Element	Yes
PAYMENT+	N/A	Payment Transaction Element	Yes

<TRANSACTION type=Return>

Element Name	Data Type/Value	Description	Required
transactionDate	YYYYMMDDHHMMSS	Date of the transaction.	Yes
transactionNumber	Number	A unique number used to identify each transaction.	Yes
subTransactionType	RETD, EXCHI, DEXCHI	Denotes the type of Return transaction. RETD – Disposed Return EXCHI – Exchange In (exchange return) DEXCHI – Disposed exchange in	No
storeNumber	String	The Store number that the transaction will be attributed to.	Yes
customerOrderNumber	String	The Customer Order that generated the transaction.	Yes
customerOrderType	S, E	Order Type values include “S” for Standard and “E” for Employee	Yes
customerOrderSource	T, M, I, G	Order source of the current order. Current values included Telephone, Mail Order, Internet, Gift Registry	Yes
customerOrderDate	YYYYMMDDHHMMSS	Create date of the order.	Yes
bannerCode	String	The banner code that is associated to the customer order header.	Yes
mediaCode	String	The media code that is associated to the customer order header. This can be different than the media code at the line level.	Yes
orderAcceptanceUser	String	The user name of the user that initiated the sale.	Yes

Element Name	Data Type/Value	Description	Required
employeeNumber	String	For RCOM XI, this value will reflect 1 for an employee sale (and mail order order type) and a –1 for a standard order.	Yes
returnRequiredFlag	Y/N	Flag indicating if return of merchandise is required by the customer.	Yes
CUSTOMER	N/A	Customer Transaction Element	Yes
SERVICE*	N/A	Service Transaction Element	No
ITEM+	N/A	Item Transaction Element	Yes
PAYMENT+	N/A	Payment Transaction Element	Yes

<TRANSACTION type=PaidIn>

Element Name	Data Type/Value	Description	Required
transactionDate	YYYYMMDDHHMMSS	Date of the transaction.	Yes
transactionNumber	Number	A unique number used to identify each transaction.	Yes
reasonCode	ACCT, PTGW, 1	Code identifying reason for the paid-in transaction. ACCT – Physical tender approval PTGW – Payment to Good Will 1 – Replacement In	Yes
storeNumber	String	The Store number that the transaction will be attributed to.	Yes
customerOrderNumber	String	The Customer Order that generated the transaction.	Yes

Element Name	Data Type/Value	Description	Required
customerOrderType	S, E	Order Type values include “S” for Standard and “E” for Employee	Yes
customerOrderSource	T, M, I, G	Order source of the current order. Current values included Telephone, Mail Order, Internet, Gift Registry	Yes
customerOrderDate	YYYYMMDDHHMMSS	Create date of the order.	Yes
bannerCode	String	The banner code that is associated to the customer order header.	Yes
mediaCode	String	The media code that is associated to the customer order header. This can be different than the media code at the line level.	Yes
orderAcceptanceUser	String	The user name of the user that initiated the sale.	Yes
employeeNumber	String	For RCOM XI, this value will reflect 1 for an employee sale (and mail order order type) and a –1 for a standard order.	Yes
CUSTOMER	N/A	Customer Transaction Element	Yes
PAYMENT	N/A	Payment Transaction Element	Yes

<TRANSACTION type=PaidOut>

Element Name	Data Type/Value	Description	Required
transactionDate	YYYYMMDDHHMMSS	Date of the transaction.	Yes
transactionNumber	String	Transaction identifier for the transaction.	Yes
subTransactionType	EXCH	EXCH – Exchange (exchange sale cancellation)	Yes
reasonCode	OVPY, CSTTAX, CSTGEN, CANSAL, CANCEL, CSTRFD, CASHOU, 2	Code identifying reason for the paid-out transaction. OVPY – Physical tender overpayment CSTTAX – Customer tax accommodation CSTGEN – Customer general accommodation (order total) CANSAL – Exchange sale cancel CANCEL – Normal order line cancel CSTRFD – Customer refund for return CASHOU – Stored value cash out 2 – Replacement Out	Yes
storeNumber	String	The Store number that the transaction will be attributed to.	Yes
customerOrderNumber	String	The Customer Order that generated the transaction.	No Not required for Stored Value Card CASHOUT
customerOrderType	S, E	Order Type values include “S” for Standard and “E” for Employee	No Not required for Stored Value Card CASHOUT

Element Name	Data Type/Value	Description	Required
customerOrder Source	T, M, I, G	Order source of the current order. Current values included Telephone, Mail Order, Internet, Gift Registry	No Not required for Stored Value Card CASHOUT
customerOrder Date	YYYYMMDDH HMMSS	Create date of the order.	No Not required for Stored Value Card CASHOUT
bannerCode	String	The banner code that is associated to the customer order header.	Yes
mediaCode	String	The media code that is associated to the customer order header. This can be different than the media code at the line level.	No Not required for Stored Value Card CASHOUT
orderAcceptanc eUser	String	The user name of the user that initiated the sale.	No Null for CASHOUT
employeeNumb er	String	For RCOM XI, this value will reflect 1 for an employee sale (and mail order order type) and a –1 for a standard order.	No Null for CASHOUT
CUSTOMER	N/A	Customer Transaction Element	Yes
PAYMENT	N/A	Payment Transaction Element	Yes

<TRANSACTION type=Dclose>

Element Name	Data Type/Value	Description	Required
transactionDate	YYYYMMDDHHMMSS	Date of the transaction.	Yes
storeNumber	String	The Store number that the transaction will be attributed to.	Yes
bannerCode	String	The banner code that is associated to the customer order header.	Yes

Element Name	Data Type/Value	Description	Required
fileCount	String	Number of files exported for sales audit for the transaction date.	Yes

<CUSTOMER > Element

Attribute Name	Data Type/Value	Description	Required
id	String	The customer identifier that the customer order is associated to.	Yes
type	B, S	The type of customer address, either it will be a bill-to or a ship-to address.	Yes

Element Name	Data Type/Value	Description	Required
firstName	String	The first name associated to the customer for this address.	Yes
middleInitial	Char(1)	The middle initial associated to the customer for this address.	No
lastName	String	The last name associated to the customer for this address.	Yes
addressLine1	String	The first line associated to the customer address.	Yes
addressLine2	String	The second line associated to the customer address.	No

Element Name	Data Type/Value	Description	Required
addressLine3	String	The third line associated to the customer address.	No
city	String	The city associated to the customer address.	Yes
state	String	The state associated to the customer address.	Yes
countyName	String	The county associated to the customer address.	Yes
postalCode	String	The postal code associated to the customer address.	Yes
countryCode	String	The country code associated to the customer address.	Yes
dayTelephoneNumber	String	The phone number associated to the customer address.	No
eveningTelephoneNumber	String	The evening phone associated to the customer address.	No
emailAddress	String	The primary email associated to the customer address.	No

<ACCOMMODATION> Element

The following table outlines those elements that can appear within the <ACCOMMODATION> element.

Attribute Name	Data Type/Value	Description	Required
id	String	The customer identifier that the customer order is associated to.	Yes

Element Name	Data Type/Value	Description	Required
accommodationType	M, G, P, H, T, O	Denotes the type of accommodation M – Merch G – Gifting P – Personalization H – S&H T – Tax O – Order Total accommodation	Yes
accommodationAmount	AMOUNT	Accommodation amount	Yes
accommodationReason	T, L, D, TL, DL, S	The reason code for the accommodation. Used for Accommodation type discounts. T – Tax Credit (order level) L – Late (order level) D- Damaged (order level) TL – Tax Credit (order line level) DL – Damaged (order line level) S - Scratched (order line level)	Yes
TAX?	N/A	Accommodation tax credit amount (for post-sale general accommodation).	No
ITEM?	N/A	Accommodation Item	No

<SERVICE> Element

The following table outlines those elements that can appear within the <SERVICE> element.

Element Name	Data Type/Value	Description	Required
serviceType	STDSH, ADDSH, RUSHSH, GIFTWRAP, GIFTCARD, PERS, MONO, RETURNPOSTAGE	Services applied to the order. STDSH – Standard S&H ADDSH – Additional S&H RUSHSH – Rush S&H GIFTWRAP – Gift Wrap GIFTCARD – Gift Card PERS – Personalization MONO – Monogramming RETURNPOSTAGE – Return postage	Yes
serviceQuantity	QUANTITY	The number of items that will be receiving the service. If the service is S&H, the quantity will always be 1, regardless of how many items were shipped as part of this transaction.	Yes
serviceUnitAmount	AMOUNT	Service amount by unit (total service charge will be unitAmount x serviceQuantity).	Yes
DISCOUNT*	N/A	Discount(s) associated to the service	No
TAX?	N/A	Tax associated to the service. The Tax element will only be available for STDSH and RUSHSH service elements. All other tax is held at the Item level for an order line.	No
ITEM?	N/A	Service Item associated to service – See note below.	No

Note: The ITEM element present on the SERVICE portion of the transaction will only be populated on the XML once per transaction in the case of Standard Shipping and Handling (serviceType = STDSH). The ITEM element will contain no qty or financial information and will be for RTLOG creation information only.

<ITEM> Element

The following table outlines those elements that can appear within the <ITEM> element.

Attribute Name	Data Type/Value	Description	Required
type	COMP, GIFT, PACK, PART, REG, BOM, LABOR, RAW, SPEC, SWATCH, GCN		

Element Name	Data Type/Value	Description	Required
itemNumber	String	Inventory Item Number	Yes
itemNumberType			
mediaCode	String	Media Code for the Customer Order Line or Service Line.	Yes
sellingItemNumber	String	The selling item number from the media	Yes
customerOrderLineNumber	String	The associated customer order line number that generated this transaction.	No
itemQuantity	QUANTITY	The number of units sold.	Yes
sellingUnitOfMeasure			
sellingUnitPrice	Number	The selling price of the item per unit.	Yes
suggestedRetailPrice			
taxableFlag	true / false	Indicates if the item is taxable.	Yes
itemMerchFlag			
directShipFlag	true / false	Indicates if the item was a direct ship item. True indicates it was a direct ship item.	No

Element Name	Data Type/Value	Description	Required
directShipSupplierID	String	Supplier ID for direct ship items. If the directShipFlag is false, this field will be null.	No
orderAcceptanceUser	String	The user name of the user that initiated the sale.	Yes
giftCertificateNumber	String	The control number associated to the gift certificate or gift card that was sold.	No
giftCertificateType	CERT, CARD	Indicates if a gift certificate or gift card was sold.	No
giftCertificateExpDate	String	Indicates the experation date of the gift certificate	No
giftCertificateToName	String	Indicates the TO name that the gift certificate is being sent to.	No
giftCertificateToCountryCode	String	Country code of ship to address	No
returnReasonCode	String	If the ITEM element is associated to a Return Transaction, this field will contain the reason why the customer returned the item.	No
ACCOMMODATION*	N/A	Accommodation Element	No
DISCOUNT*	N/A	Discount Element	No
SERVICE*	N/A	Service Element	No
TAX	N/A	Tax Element	Yes

<PAYMENT> Element

The following table outlines those elements that can appear within the <PAYMENT> element.

Attribute Name	Values	Description
id	String	Unique payment identifer
type	Cash, Check, CreditCard, GiftCert, MerchCert, GiftCard, MerchCard, RewardCert, CustGoodwill	Cash – Cash payment Check – Check payment CreditCard – Credit card payment GiftCert – Gift certificate payment MerchCert – Merchandise certificate payment GiftCard – Stored value Gift card payment MerchCard – Stored value merchandise card payment RewardCert – Reward certificate payment CustGoodwill – Customer goodwill payment, used when a customer order is underpaid but within the tolerance defined in the system.

<PAYMENT type = Cash> Element

Element Name	Data Type/Value	Description	Required
tenderTypeId	String	Unique RCOM Tender Type ID	Yes
tenderTypeGroupId	String	Unique TenderType Group ID	Yes
paymentAmount	AMOUNT	Amount Element	Yes

<PAYMENT type = Check> Element

Element Name	Data Type/Value	Description	Required
tenderTypeId	String	Unique RCOM Tender Type ID	Yes
tenderTypeGroupId	String	Unique TenderType Group ID	Yes
paymentAmount	AMOUNT	Amount Element	Yes
checkAccountNumber	String	Check account number	No

Element Name	Data Type/Value	Description	Required
checkRoutingNumber	String	Check routing number.	No
checkCheckNumber	String	Check number	No
checkAuthorizationCode	String	Not used	No
checkAuthorizationDate	String	Not used	No

<PAYMENT type = CreditCard> Element

Element Name	Data Type/Value	Description	Required
tenderTypeId	String	Unique RCOM Tender Type ID	Yes
tenderTypeGroupId	String	Unique TenderType Group ID	Yes
paymentAmount	AMOUNT	Amount Element	Yes
creditCardAuthoriziationDate	YYYYMMDDHHMMSS	Timestamp payment was authorized via credit approval service	Yes
creditCardExpirationDate	YYYYMM	Card expiration date (YYYY-MM)	Yes
creditCardAccountNumber	Encrypted? Masked?	The credit card account number.	Yes
creditCardCardHolderFirstName	String	Card holder first name	Yes
creditCardCardHolderMiddleInitial	String	Card holder middle initial	No
creditCardCardHolderLastName	String	Card holder last name	Yes
creditCardRespAddressVerificationCode	String	The AVS code returned from the credit approval service.	No

Element Name	Data Type/Value	Description	Required
creditCardRespAuthorizationCode	String	The authorization code returned from the credit approval service.	No
creditCardRespCVVVerificationCode	String	The CVS code returned from the credit approval service.	No
creditCardReferenceField1	String	Reference field	No
creditCardReferenceField2	String	Reference field	No
creditCardReferenceField3	String	Reference field	No
creditCardReferenceField4	String	Reference field	No
creditCardReferenceField5	String	Reference field	No
creditCardReferenceField6	String	Reference field	No
creditCardReferenceField7	String	Reference field	No
creditCardReferenceField8	String	Reference field	No
creditCardReferenceField9	String	Reference field	No
creditCardReferenceField10	String	Reference field	No

<PAYMENT type = GiftCert/MerchCert/GiftCard/MerchCard/RewardCert> Element

Element Name	Data Type/Value	Description	Required
tenderTypeId	String	Unique RCOM Tender Type ID	Yes
tenderTypeGroupId	String	Unique TenderType Group ID	Yes
paymentAmount	AMOUNT	Amount Element	Yes
voucherNumber	Number	The Control number for the voucher. This attribute will be set for sale payments, credit payments in RCOM do not carry this attribute.	No

<PAYMENT type = CustGoodwill,Merchandise> Element

Element Name	Data Type/Value	Description	Required
tenderTypeId	String	Unique RCOM Tender Type ID	Yes
tenderTypeGroupId	String	Unique TenderType Group ID	Yes
paymentAmount	AMOUNT	Amount Element	Yes

<TAX> Element

The following table outlines those elements that can appear within the <TAX> element.

Element Name	Data Type/Value	Description	Required
taxAmount	AMOUNT	Amount of tax applied	Yes
taxCode	String	Tax code to represent whether the tax represents a state tax type, provincial tax, etc (Code type of TAXC)	Yes
merchandiseTaxAmount	String	Merchandise Tax for transaction	No
shippingTaxAmount	String	Shipping & Handling Tax for transaction	No
giftingServiceTaxAmount	String	Gift service tax amount	No
personalizationServiceTaxAmount	String	Personalization service tax amount	No

<DISCOUNT> Element

The following table outlines those elements that can appear within the <DISCOUNT> element.

Element Name	Data Type/Value	Description	Required
discountType	SS, AS, PR, MO, GW, GC, GE, IT, LM, OM, PC, SL, EM	Denotes the type of discount. SS – Standard Shipping promotion AS – Additional Shipping promotion PR – Personalization promotion	Yes

Element Name	Data Type/Value	Description	Required
		MO – Monogramming promotion GW – Gift wrap promotion GC – Gift card promotion GE – Gift certificate promotion IT – Item promotion LM – Line merch promotion OM – Order merch promotion PC – Plan Code promotion SL – Selling list promotion EM – Employee discount	
discountAmount	Number	The total discount amount	Yes
promotionNumber	String	If this discount is associated to a promotion the corresponding promotion ID.	No
promotionFormatType	String	G – General promotion T – Threshold promotion	No

<AMOUNT> Entity

Attribute Name	Data Type/Value	Description	Required
currency	String	USD	Yes

```
<AMOUNT> Example layout:
<paymentAmount currency = “USD”>100.00</paymentAmount>
```

<QUANTITY> Entity

```
 <QUANTITY> Example layout:
<itemQuantity>10.0</itemQuantity>
```

ReSA mapping – all RCOM-related transactions

The following table represents a consolidated list of the data that RCOM is sending to ReSA for sales, returns, exchanges, accommodations, and replacements.

Record Name	Field Name	Field Type	Default Value	Description	Required	Justification/ Padding
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number, starting at 1.	ID of current line being processed by input file.	Y	Right/0
	File Type Definition	Char(4)	RTLG	Identifies file as 'Retek TLOG'.	Y	Left/Blank
	File Create Date	Char(14)	System date	Date and time file was written by external system (YYYYMMDDHHMMSS).	Y	Left/None
	Business Date	Char(8)	Date of earliest transaction in the file.	Business date of transactions. (YYYYMMDD).	Y	Left/None

Record Name	Field Name	Field Type	Default Value	Description	Required	Justification/ Padding
	Location Number	Char(10)	SALE/RETURN: Store id associated to media code associated to order line PAIDIN: associated to the store on the order PAIDOU: associated to the store on the order ACCOMMODATIONS: VAS and S&H discount will be associated to the store at the order header level, the merch discount will be associated to the order line store.	Store identifier.	Y	Left/None
	Reference Number	Char(30)	blanks		N	Left/Blank

Transaction Header	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Register	Char(5)	00000	Till used at store.	Y	Left/Blank
	Transaction Date	Char(14)	For sales: ship_container.shipped_date For returns: date of WMS sending return to RCOM For accommodations: date will vary depending on when accommodation is sent For paidin/paidou: date that transaction is staged for RESA In YYYYMMDDHHMMSS format	Date transactions were processed at the POS (YYYYMMDDHHMMSS).	Y	Left/None

	Transaction Number	Number(10)	Sequential incrementing transaction number per RTLOG file. Will be empty for DCLOSE.	Transaction identifier.	Y	Right/0
	Banner id	Number(4)	Banner associated to customer order	The unique identifier of the banner.	Y	
	Media id	Char(10)	Media code associated to customer order header.	The identifier of the media	N	
	Customer Order Head No	Char(30)	customer order that the transaction is associated to	The identifier of a customer order line.	N	
	Customer Order Head Date	Char(14)	use customer_order.create_date	The customer order creation date	N	
	Cashier	Char(10)	User id for order creation	Cashier identifier.	N	Left/Blank
	Salesperson	Char(10)	User id for order creation	Salesperson identifier.	N	Left/Blank

	Employee ID	Char(10)	Employee ID. In the short term RCOM will be sending a value of 1 if the transaction is an employee sale (or supporting transaction). A '-1' will indicate a normal sale.	Employee identifier.	N	Left/Blank
	Transaction Type	Char(6)	For Sale: 'SALE' For Returns: 'RETURN' For Exchange Return: 'RETURN' For Exchange Sale: 'SALE'	Transaction type.	Y	Left/Blank

	Sub-transaction type	Char(6)	For Return with disposed disposition: 'RETD' For non-disposed exchange return: 'EXCHI' For disposed exchange return: 'DEXCHI' For exchange sale: 'EXCHO' For PAIDOU for the exchange sale cancellation refund: 'EXCH'	Sub-transaction type. For sale, it can be employee, drive-off etc.	N	Left/Blank
	Orig_tran_no	Number(10)	Blank	Populated only for post-void transactions. Transaction number for the original tran that will be cancelled.	N	Right/0
	Orig_reg_no	Char(5)	blank	Populated only for post-void transactions. Register number from the original tran.	N	Left/Blank

	Reason Code	Char(6)	PAIDIN of physical tender on order: 'ACCT' Refund PAIDOU: 'OVPY' Tax accommodation PAIDOU: CSTTAX General accommodation PAIDOU: CSTGEN PAIDIN to cover underpayment: 'PTGW' Replacement in: 1 Replacement out: 2 For all other trans types: blanks	Reason entered by cashier for some transaction types. Required for Paid In and Paid out transaction types, but can also be used for voids, returns, etc.	N	Left/Blank
	Vendor Number	Char(10)	blank	Supplier id for a merchandise vendor paid out transaction, partner id for an expense vendor paid out transaction.	N	Left/Blank
	Vendor Invoice Number	Char(30)	Blank	Invoice number for a vendor paid out transaction.	N	Left/Blank

	Payment Reference Number	Char(16)	Blank	The reference number of the tender used for a vendor payout. This could be the money order number, check number, etc.	N	Left/Blank
	Proof of Delivery Number	Char(30)	Blank	Proof of receipt number given by the vendor at the time of delivery. This field is populated for a vendor paid out transaction.	N	Left/Blank

	Reference Number 1	Char(30)	FOR DCLOSE: The number of files sent to RESA for a given store/day. For example, if there were 3 files for store X, the first two files would not have a DCLOSE record but the third and final file would have a DCLOSE record. For the third file with the DCLOSE, the value in ref_no1 would be 3. FOR PAIDIN: account number of MC/GC FOR RETURN: receipt indicator 'Y' or 'N'	Number associated with a particular transaction, for example weather for a Store Conditions transaction. The sa_reference table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference Number 2	Char(30)	RCOM populates with physical store number, if available.	Second generic reference number.	N	Left/Blank

	Reference Number 3	Char(30)	For Sales: order number associated to transaction For Returns: order number of return order's original order	Third generic reference number.	N	Left/Blank
	Reference Number 4	Char(30)		Fourth generic reference number.	N	Left/Blank
	Value Sign	Char(1)	'P' for positive, 'N' for negative blank for day-end	Sign of the value.	Y if Value is present	Left/None
	Value	Number(20)	Total selling price of all TITEM records + total tax on all TITEM records for this transaction For day end: blanks	Value with 4 implied decimal places. Populated by the retailer for TOTAL trans, populated by Retek sales audit for SALE, RETURN trans.	Y if tran is a TOTAL.	Right/0 when value is present. Blank when no value is sent.

Transaction Customer	File Type Record Descriptor	Char(5)	TCUST	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number (10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Customer ID	Char(16)	Customer.customer_id	The ID number of a customer.	Y	Left/Blank
	Customer ID type	Char(6)	CUSTID	Customer ID type.	Y	Left/Blank
	Customer Name	Char(40)	A ship container is associated to a ship request, take the customer_id from ship_request and look at the customer_address table to get the customer name. The bill-to address info will be from the address table. The first_name, middle_initial, last_name will have to be joined	Bill-to Customer name.	N	Left/Blank
	Address 1	Char(40)	Address.Line1	Customer bill-to address.	N	Left/Blank
	Address 2	Char(40)	Address.line2	Additional field for customer Bill-to address.	N	Left/Blank
	City	Char(30)	Address.city	Bill-to City.	N	Left/Blank
	State	Char(3)	Address.state	Bill-to State.	N	Left/Blank
	Zip Code	Char(10)	Address.postal_code	Bill-to Zip code.	N	Left/Blank

	Country	Char(3)	Address.country_code	Bill-to Country.	N	Left/Blank
	Home Phone	Char(20)	Customer_telephone.primary_evening_telephone_flag	Telephone number at home.	N	Left/Blank
	Work Phone	Char(20)	Customer_telephone.primary_day_telephone_flag	Telephone number at work.	N	Left/Blank
	E-mail	Char(100)	Customer_email.email_address for customer_id associated to ship_request	E-mail address.	N	Left/Blank
	Birthdate	Char(8)	blank	Date of birth. (YYYYMMDD)	N	Left/Blank

Customer Attribute	File Type Record Descriptor	Char(5)	CATT	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Attribute type	Char(6)	Refer to ‘SACA' code_type for a list of valid types	Type of customer attribute	Y	Left/Blank
	Attribute value	Char(6)	Refer to members of ‘SACA' code_type for a list of valid values	Value of customer attribute.	Y	Left/Blank

Transaction Item	File Type Record Descriptor	Char(5)	TITEM	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Item Status	Char(6)	'S' for sale, exchange sale 'R' for return, exchange return	Status of the item within the transaction, V for item void, S for sold item, R for returned item.	Y	Left/Blank

	Item Type	Char(6)	If the item_type = 'REG' and the merchandise_flag is YES then the item_type sent to RESA is 'ITEM'. If the item_type = 'REG' and the merchandise_flag is NO in item_master, then the item_type sent to RESA is 'NMITEM'. If the item_type = 'GIFT', then the item_type sent to RESA is 'GCN'.	Identifies what type of item is transmitted.	Y	Left/Blank
	Item number type	Char(6)	'ITEM' – retrieved from item_master	Identifies type of item number if item type is ITEM or REF.	N	Left/Blank
	Format ID	Char(1)	blank	Used to interpret VPLU items.	N	Left/Blank

	Item	Char(25)	= customer_order_line.selling_sku_id when item type = 'ITEM'	Identifies merchandise item.	N	Left/Blank
	Reference Item	Char(25)	Blank	Identifies sub-transaction level merchandise item.	N	Left/Blank
	Non-Merchandise Item	Char(25)	= customer_order_line.selling_sku_id when item type = 'NMITEM' Used for GC, S&H	Identifies non-merchandise item.	N	Left/Blank

	Media Line ID	Char(10)	For 'regular' items, this will be the media associated to the order line. For VAS TITEM this will be the media associated to the order line that the VAS is associated to. For S&H TITEM this will be a store in the shipped container.	The media code attached to the order line where this item was ordered.	N	Left/Blank
	Selling Item id	Char(25)	= customer_order_line.(media_code+selling_sku_id) Selling SKU number from the order line associated to the shipped container line. The media_code is from the first order line on the order.	The unique identifier of a selling item.	N	Left/Blank

	Customer Order Line No	Number(30)	Customer order line number associated to the shipped container line. FOR RETURNS: original order line number	The identifier of a customer order line.	N	
	Voucher	Char(16)	Blank For GC: GC number	Gift certificate number	N	Right/0
	Department	Number(4)	blank	Identifies the department this item belongs to. This is filled in by saimptlog.	N	Right/Blank
	Class	Number(4)	blank	Class of item sold or returned. Not required from a retailer, populated by Retek sales audit. This is filled in by saimptlog.	N	Right/Blank
	Subclass	Number(4)	blank	Subclass of item sold or returned. Not required from a retailer, populated by Retek sales audit. This is filled in by saimptlog.	N	Right/Blank
	Quantity Sign	Char(1)	'P' for positive	Sign of the quantity	Y	Left/None

	Quantity	Number(12)	FOR SALE: Shipped quantity of the selling item on the ship container line. FOR RETURN: returned qty	Number of items purchased with 4 decimal places.	Y	Right/0
	Selling Unit of Measure	Char(4)	If the Selling_UOM field is empty on item_location, then the item_master.standard_uom must be used in the RTLOG TITEM Selling_UOM field.	Unit of measure of item's quantity.	Y	Left/None
	Unit Retail	Number(20)	media_selling_sku.selling_sku_unit_price for the selling sku associated to the inventory SKU in the shipped container line.	Unit retail with 4 implied decimal places.	Y	Right/0

	Override Reason	Char(6)	'RCOM'	This column will be populated when an item's price has been overridden at the POS to define why it was overridden.	Y if unit retail was manually entered	Left/Blank
	Original Unit Retail	Number(20)	Item_master.unit_retail for the selling sku associated to the inventory SKU in the shipped container line.	Value with 4 implied decimal places. This column will be populated when the item's price was overridden at the POS and the item's original unit retail is known.	Y if unit retail was manually entered	Right/0
	Taxable Indicator	Char(1)	'Y' for yes 'N' for no look at RMM item_location table.	Indicates whether or not item is taxable.	Y	Left/None
	Pump	Char(8)	Blank	Fuel pump identifier.	N	Left/Blank
	Reference Number 5	Char(30)	Blank FOR GC: recipient name	Number associated with a particular item within a transaction, for example special order number. The sa_reference table defines what this field can contain for each transaction type.	N	Left/Blank

	Reference Number 6	Char(30)	Blank FOR GC: recipient state	Second generic reference number at the item level.	N	Left/Blank
	Reference Number 7	Char(30)	Blank FOR GC: recipient country	Third generic reference number at the item level.	N	Left/Blank
	Reference Number 8	Char(30)	Blank	Fourth generic reference number at the item level.	N	Left/Blank
	Item_swiped_ind	Char(1)	'N' for no – will be no	Indicates if the item was automatically entered into the POS system or if it had to be manually keyed.	Y	Left/None
	Return Reason Code	Char(6)	For returns, from return order line value. For sales: blank	The reason an item was returned.	N	Left/Blank
	Salesperson	Char(10)	User name of updated user on order line	The salesperson who sold the item.	N	Left/Blank
	Expiration_date	Char(8)	Blank For GC: GC expiration date (can be null)	Gift certificate expiration date (YYYYMMDD).	N	

	Drop Ship Ind	Char(1)	'Y' for yes. Will be retrieved from the order line associated to the ship container line. 'N' for no	Indicates whether item is direct ship.	Y	Left/None
	Supplier	Number(10)	Direct ship supplier associated to direct ship item on ship container line. Supplier will be on the order line associated to a direct ship item	The Direct Ship Supplier associated to the Direct Ship Ind. If the Drop Ship Ind is 'N' this field will be NULL.	N	Left/Blank

Item Discount	File Type Record Descriptor	Char(5)	IDISC	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number (10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	RMS Promotion Number	Char(6)	Accommodations: 1004 General discount promotion: 2000	The RMS promotion type.	Y	Left/Blank
	Discount Reference Number	Number (10)		Discount reference number is associated with the discount type (e.g. if discount type is a promotion, this contains the promotion number).	N	Left/Blank
	Discount Type	Char(6)	Order line VAS accommodation Discount type: CSTVAS Order line S&H accommodation Discount type: CSTSH Order line Merch accommodation Discount type: CSTMER General discount promotions: G	The type of discount within a promotion. This allows a retailer to further break down coupon discounts within the “In-store” promotion, for example.	N	Left/Blank
	Coupon Number	Char(16)	Blank	Number of a store coupon used as a discount.	Y if coupon	Left/Blank

	Coupon Reference Number	Char(16)	Blank	Additional information about the coupon, usually contained in a second bar code on the coupon.	Y if coupon	Left/Blank
	Quantity Sign	Char(1)	P	Sign of the quantity.	Y	Left/None
	Quantity	Number (12)	Qty of item that discount is applied to	The quantity purchased that discount is applied with 4 implied decimal places.	Y	Right/0
	Unit Discount Amount	Number (20)	Discount amount/item	Unit discount amount for this item with 4 implied decimal places.	Y	Right/0
	Reference Number 13	Char(30)	Blank	Number associated with a particular transaction type at the discount level. The sa_reference table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference Number 14	Char(30)	Blank	Second generic reference number at the discount level.	N	Left/Blank
	Reference Number 15	Char(30)	Blank	Third generic reference number at the discount level.	N	Left/Blank
	Reference Number 16	Char(30)	Blank	Fourth generic reference number at the discount level.	N	Left/Blank

Transaction Tax	File Type Record Descriptor	Char(5)	TTAX	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Tax Code	Char(6)	"STATE"	Tax code to represent whether it is a state tax type, provincial tax, etc.	Y	Left/Blank
	Tax Sign	Char(1)	positive: "P"	Sign of Tax Amount.	Y	Left/None
	Tax Amount	Number(20)	Total will include: Merch Tax + s&h tax + add'l delivery tax + rush delivery tax + VAS Tax Total will be prorated per item per store. 4 implied decimal places	Amount of tax charged for this tax code type in a transaction with 4 implied decimal places.	Y	Right/0
	Reference Number 17	Char(30)	Merchandise tax for transaction.	Generic reference number.	N	Left/Blank
	Reference Number 18	Char(30)	S&H Tax for transaction.	Generic reference number.	N	Left/Blank
	Reference Number 19	Char(30)	VAS Tax for transaction.	Generic reference number.	N	Left/Blank

	Reference Number 20	Char(30)		Generic reference number.	N	Left/Blank

Transaction Tender	File Type Record Descriptor	Char(5)	TTEND	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Tender Type Group	Char(6)	Tender type on order payment	High-level grouping of tender types.	Y	Left/Blank
	Tender Type ID	Number(6)	Tender type id on order payment	Low-level grouping of tender types.	Y	Left/Blank
	Tender Sign	Char(1)	‘P’ for positive	Sign of the value.	Y	Left/None
	Tender Amount	Number(20)	Total of unit retail fields from all TITIEM records plus the tax amount from TTAX record minus other TTEND Payment settlement amount.	Amount paid with this tender in the transaction with 4 implied decimal places.	Y	Right/0

	Cc_no	Number(16)	Customer_order_payment.account_number where credit_card_id is populated and order is associated to shipped container. For payment types other than CC: blanks	Credit card number	Y if credit card	Left/Blank
	Cc_auth_no	Char(16)	Customer_order_payment.authorization_code on order that is associated to shipped container. For payment types other than CC: blanks	Authorization number for a cc	Y if credit card	Left/Blank
	cc authorization source	Char(6)	For CC: 'E' for electronic – default For CC: 'M' for manual For payment types other than CC: blanks		Y if credit card	Left/Blank
	cc cardholder verification	Char(6)	For CC: 'E' For payment types other than CC: blanks		Y if credit card	Left/Blank

	cc expiration date	Char(8)	Customer_order_payment.expiration_date for order associated to shipped container For payment types other than CC: blanks	(YYYYMMDD)	Y if credit card	Left/Blank
	cc entry mode	Char(6)	For CC: 'T' For payment types other than CC: blanks	Indicates whether the credit card was swiped, thus automatically entered, or manually keyed.	Y if credit card	Left/Blank
	cc terminal id	Char(5)	blanks	Terminal number transaction was sent from.	N	Left/Blank
	cc special condition	Char(6)	For CC: 'E' For payment types other than CC: blanks		Y if credit card	Left/Blank
	Voucher_no	Char(16)	blank	Gift certificate or credit voucher serial number.	Y if voucher	Right/0
	Coupon Number	Char(16)	Blank	Number of a manufacturer's coupon used as a tender.	Y if coupon	Left/Blank

	Coupon Reference Number	Char(16)	Blank	Additional information about the coupon, usually contained in a second bar code on the coupon.	Y if coupon	Left/Blank
	Reference No 9	Char(30)	Blank For checks: check number	Number associated with a particular transaction type at the tender level. The sa_reference table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference No 10	Char(30)	Blank	Second generic reference no at the tender level.	N	Left/Blank
	Reference No 11	Char(30)	Blank	Third generic reference no at the tender level.	N	Left/Blank
	Reference No 12	Char(30)	Blank	Fourth generic reference no at the tender level.	N	Left/Blank

Transaction Trailer	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number (10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	Transaction Record Counter	Number (10)	Number of lines/records between THEAD and TTAIL	No of records processed in current tran (only records between trans head & tail)		

File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type	Y	Left/Blank
	File Line Identifier	Number(10)	Sequential incrementing line number.	ID of current line being processed by input file.	Y	Right/0
	File Record Counter	Number(10)	Number of lines/records between FHEAD and FTAIL	No of transactions processed in current file (only records between file head & tail)	Y	Right/0