ARTS POSLog V6.0

Narrative Documentation

February 10, 2014 – Last call working draft

Chairman:

Julia Bond	Toshiba Global Commerce Solutions

Authors:

Rony Lindner	NCR
Cory Tiedeman	American Multi-Cinema
Christo Smit	Argility
David Chaffey	Avery Berkel
Guenther Falter	Bizerba
Gerhard Schatz	Bizerba
Perry Kramer	Boston Consulting Group
Kirstin Wright	Cumulus Data Services
Francisca Vicente Tamarin	El Corte Inglés
Leonid Rubahkin	Epicor
H. Paul Gay	Epson
Tadashi Furuhata	Epson
Hideo Nakamura	Epson
Angela Seidel	GK Software
Victor Swahn	H&M Hennes & Mauritz AB
Johan Tuvstedt	H&M Hennes & Mauritz AB
Brian Taylor	JDA Software Group
Vinay Kumar	Mercury Consulting
Werner Engeln	Mettler Toledo
Moin Moinuddin	Microsoft
Charlie Souhrada	NAFEM
Jeff Longino	NCR
Pavan Rudraraju	Nike
Morten Folvell	NorgesGruppen
Dave Moorman	PCMS Datafit
Matthew Flower	PCMS Group
Graham Hill	Post Office UK
Sharon Dweck	Retalix
Brian Taylor	SofTechnics

Dave Parr	Unipower Solutions
Rajul Garg	WIPRO
George Burnett	Yum! Brands
Richard Halter	ARTS
Tom Sterling	ARTS

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

1.1 Why Create POSLog V6?

The first version of POSLog was published in 1999. Initially, it was simply a migration of the old TLog model into XML. But over time it became apparent that POSLog was much more valuable to all of retail beyond just as a log. The sales and financial information contained within POSLog lies at the heart of retailing, and virtually every system is somehow dependent upon this information.

ARTS built other variants of POSLog to support specific uses (Foodservice, RTI to expose transaction data as a service), but it became apparent we needed a more flexible POSLog architecture to support these different uses without burdening them with all the extraneous information needed for the other variants.

For example, RTI supports sharing item, tender and tax information between multiple touchpoints, such as kiosks or phones with a server. But the previous POSLog architecture saddled this model with components such as Tender Control Information used for reporting non-sales tenders functions. The same problem is true of other POSLog alternatives such as Digital Reciept, Price, Tax and Customer.

POSLog V6 was architected to solve this problem. ARTS can now produce a stripped down version of POSLog to satisfy specific sharing needs, as well as a complete version to handle the traditional uses. Both versions will now be compliant on the same set of information. Obviously additional information in the complete version will not validate in the stripped down version, but all information in the stripped down version will validate against the complete version.

1.2 POSLog — the Heart of Retailing

In the olden early days of retail, stores started tracking sales by handwriting receipts. As electronics became more involved with the process, handwritten records migrated to electronic journals and a de facto standard known as TLog arrived. This TLog was electronically recorded and, at the end of the day, it was uploaded to corporate systems for processing. Its numerous uses included tracking inventory and balancing the tender information with the deposits. The primary focus of this evolution was to record everything that happened at the Point of Sale.

The next phase of the evolution, which still continues to this day, was to integrate all systems in the store to provide real-time information about sales, inventory and tender. With new architectures, this evolution is going to what could be termed micro-real-time. Instead of the traditional transaction being published for use by these connected systems, the new breed of service-oriented architectures are requiring the individual components of a transaction to be available as soon as they are entered into the POS. The scope covers all transactions – mobile, e-commerce, kiosks, call centers and stores.

This is where the ARTS Standard POSLog comes into play. POSLog provides the message set necessary to transmit Point of Service information to anywhere it's needed in the Enterprise and beyond. This widely-adopted, powerful interface has revolutionized the information flow within an enterprise. POSLog enables the new service-oriented architectures to have access to this micro-real-time item information and it does it without sacrificing the traditional end of day batch processing or real-time transaction messages.

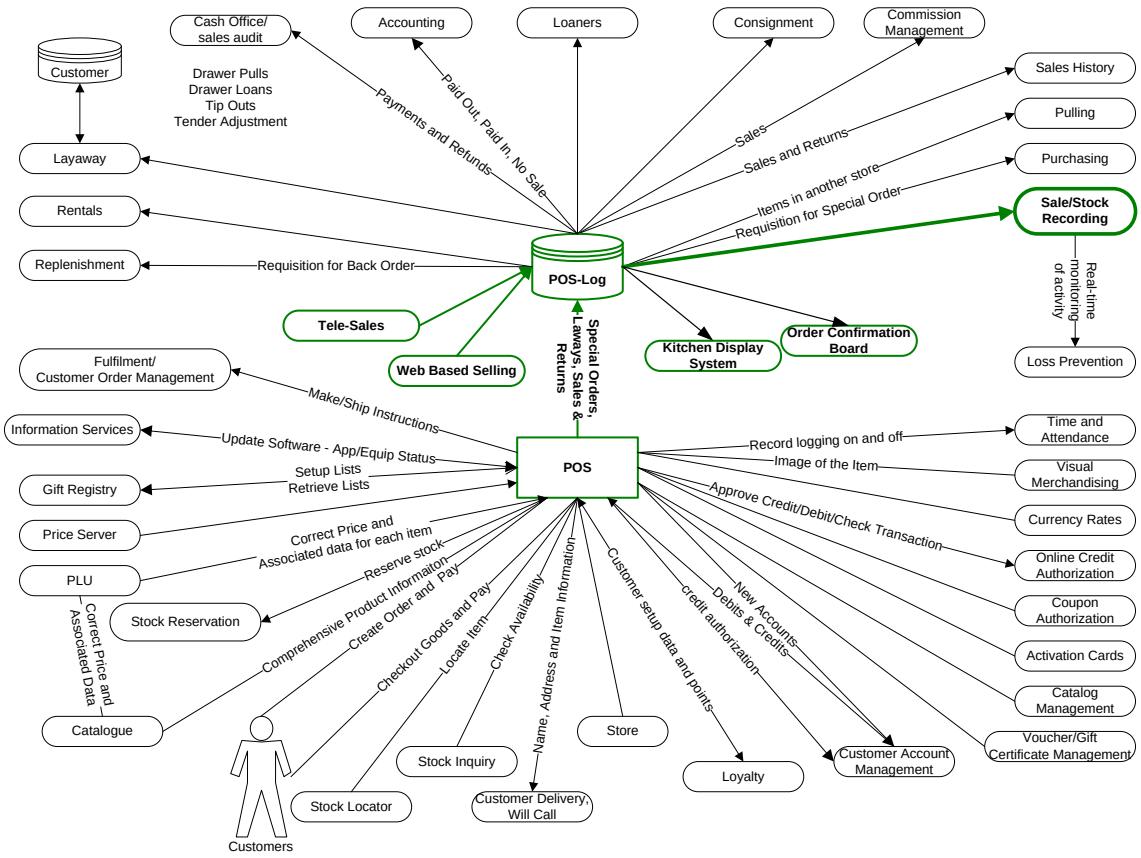


Figure 1: Retail Model for POS

1.3 Real-Time, End-of-Day and Batch Process Models

During the day the POS provides three major categories of information. The first is what everyone expects — normal transactional data. In addition there is a host of non-transactional financial data, for example money is taken out of the till to pay for a window washer, or money is taken in from the newspaper vendor. This type of POSLog transaction is called a Tender Control Transaction. The third major set of information deals with real-time feeds to areas such as Store Operations or Loss Prevention to support timely decision making. Additionally, POSLog supports two important areas, tender control and event information, as well as a few minor targeted areas dealing with inventory control, foodservice and forecourt.

The transactional component of POSLog supports the three major reporting methodologies; first the traditional batch end of day upload capability, second the traditional complete transaction and finally the individual components of a transaction.

1.3.1 Batch reporting

For batch reporting POSLog can handle several different options. The first and most granular is the aggregation of all the individual Transaction Level Detail (TLD) for every transaction that occurred at the store. The next level of granularity is the summarization of the TLD by item. For example, store 100 sold 55 blue shirts on 10/27/2006 for $250. The lowest level of granularity reports this store sold $10,000 worth of goods today. If so desired, XML schemas allow the mixing of each of these levels of granularity in one document.

In a normal enterprise this summary information has several uses. The first is the normal end-of-day processing. The middle level of granularity gives support for an Extract, Transform, Load (ETL) capability between the online transaction processing (OLTP) data base to the online analytical processing (OLAP) data warehouse. Real-time summary level information can also be used to monitor store performance through a dashboard or alerting environment.

1.3.2 Transaction reporting

The core of all store information is contained within the Transaction Level Detail of a transaction. POSLog provides extensive support for every step through the transaction life cycle. A transaction begins when the first item is scanned at the checkout register or the first item is selected in the browser or app. It then progresses as more items are added or removed. The transaction is tendered. Finally the items are delivered to the customer. Delivery occurs in a variety of different ways: the customer picks up their bags and walks out the store, the items purchased at the store are delivered to their front door, or the postal service delivers the items.

These are all significant steps in the life cycle of a transaction. To support this, ARTS has defined a transaction to have three major transactions. The first, called an Order Transaction, occurs when a shopper starts putting things into their shopping basket. This occurs prior to the checkout process. The second,

called the Cusomter Order Transaction, occurs before the transaction is tendered and delivered. The third, called a Retail Transaction, occurs after the transaction is tendered and delivered. Most enterprises will recognize the Retail Transaction as the traditional receipt given to the customer at the checkout stand. Typically the information contained in the Retail Transaction is uploaded to corporate databases at the end of the day or in real-time through trickle polling. It is used to manage the store by populating various databases such as inventory and financial accounts, and to analyze store performance and merchandising effectiveness. It is truly the lifeblood of the enterprise.

1.3.3 Shopping Basket

POSLog not only provides transactional information but it has all the information needed to populate a “shopping basket”. POSLog has migrated to become the payload wherever we need to provide a “shopping basket” for evaluation. For example, to price a transaction or calculate taxes, one needs the shopping basket. To evaluate customer loyalty, one needs the items and the customer information, and to send information to the kitchen system one needs the items and any build or cooking instructions. All this information is built into the POSLog schema.

1.3.4 Forecasted Information

The POSLog Retail Transaction has the resources to not only report what has occurred but it can be used to send “Projection” information to other applications. The forecasting system can take the Retail Transaction data, both current and historical, TLD or summary and turn it into a forecast for future needs. The forecast can then be sent to systems like the merchandising or labor scheduling system through POSLog. Merchandising can then compare the “Projections” with current inventory to make purchase decisions. Or the Labor Scheduling system can produce a better schedule based on a variety of associate criteria along with projected sales.

1.3.5 SOA Services

To really respond to today's customer, information is needed in micro-real-time. That is, information is needed as soon as it is entered in the POS. Inventory needs it to make real-time inventory reservations. Customer Relationship Management needs it to evaluate offers for platinum customers. Managers need it to resolve issues as soon as they rise. In a web environment, this could extend all the way to the supplier to retrieve and to report an inventory count on availability to the customer.

To support this POSLog has the capability to send individual line items one at a time as soon as they are entered into the POS. There are three distinct points in the development of a transaction when information can be sent. The first technique is called On-The-Fly. In this mode as each item is entered, the item informatioon is immediately presented to other systems in the store or web.

The second technique is called One-Behind. Here the first item is entered into the POS (or web). The next item is entered. At this point the first item is made available to other systems in the store. This is one of the major methods used in foodservice. It allows one to enter an item, make changes then when one moves to the next item, the first item is made available.

The next method is On-Total. Here all the items are entered into the POS (web) and when the total key is pushed, the entire list of items are presented to the other systems. Normally systems which use On-The-Fly or One-Behind send a Totalled Transaction when they have finished taking in the list of items. For example, a Kiosk might use POSLog to take an order to be sent to a checkout counter where the customer pays for the items.

There are multiple uses for this micro-real-time data. For example, in a foodservice department, one can start cooking the various line items earlier thus improving the customer experience by improving the speed of service. Or the line items can be sent to the warehouse to begin pulling the couch for the customer to pickup as soon as they get to the dock. Or one can leverage a single POS application to support multiple kiosks or a web site.

1.4 Referenced Documents

International System of Units

- Produced and managed by the Bureau International des Poids et Mesures located at http://www.bipm.fr

ARTS XML

- ARTS XML POSLog Charter, Version 2.00
- ARTS XML Extending Schemas Technical Report
- ARTS XML XML Best Practices, 2001-7-31
- ARTS XML XML Dictionary, Version 3.0 Build 04
- ARTS Notification Event Architecture for Retail Technical Specification V1.0.0 2006-08-11

International Standards Organization

- ISO 7812-7816 Based Card Numbers and the Domain Name System (DNS)
- ISO-8601: Time and Date formats
- ISO-3166-1: Country Codes
- ISO-4217: Currency Codes
- ISO-639-1: Language Codes

Tax

- Easing the impact of VAT: Consultation on a flat rate scheme for small firms, H M Customs & Excise June 2001
- Multi-State Sales and Use Tax Manual, Doris C. Locks

2. XML ARCHITECTURE DECISIONS

2.1 Optional Elements

When POSLog started there was significant discussion about including optional elements; after all how can one have a standard when almost everything is optional? ARTS investigated the alternative of having a schema for every potential use case but this was impossible, as we could never identify every possible permutation. So the alternative was to create a schema which can support all kinds of use cases but to make the elements optional.

The question then became how two systems that can interoperate as one might use this set of elements, and another use a different set of elements. Both would be valid against the schema but not be able to interoperate. The answer was in the Use Case development methodology employed by ARTS.

With each Use Case ARTS includes an exampleXML Instance Document. In each example XML Instance, ARTS has identified the minimal set of elements necessary to support this use case. Now if two applications want to “sell a shirt”, they will both know what the minimal set is of elements necessary to interoperate.

2.2 ARTS **Venetian Blind Architecture**

There are basically three architectures for building XML Schemas.

2.2.1 Russian Doll Design

At one end of the spectrum is the Russian Doll Design. As its name implies it is like a Russian Doll with one doll nested inside another. In XML terms it has a heavy use of embedded anonymous complex types. It makes everything local in scope and reuse impossible.

2.2.2 Salami Slice Design

At the other end of the spectrum is the Salami Slice Design. As one can envision with a salami slice, all elements are global. With all elements global in scope every name must be unique. It maximizes reuse but makes for a complicated schema to follow and understand.

2.2.3 Venetian Blind Design

In the middle of the spectrum is the Venetian Blind Design. Here appropriate elements are embedded within context and global complex types are defined for maximum reuse. This is a building block approach. One assembles a reuseable set of global named complex types, then attaches them to the elements defined within context.

The benefits of this approach:

1. Maximizes reuse. The primary components of reuse are the complex type definitions. These complex types are the foundation for the library of ARTS XML components used to build all ARTS schemas.
2. Maximizes cohesiveness – the complex types group together related data.
3. Minimizes coupling

POSLog Schema, as are all ARTS schemas, is built following this Venetian Blind approach. The POSLog schema contains numerous complex types which comprise the building blocks.

2.3 SOA Architecture

POSLog has been extended to support service-oriented architecture by wrapping POSLog with a set of services. This is published under the Retail Transaction Interface standard.

2.4 Extension Points

ARTS extension policy states that each work team will put extension points at places they deem critical. These are typically places, in the opinion of the work team, where it is possible to add other vendor-relevant information.

One of those places is in one of the top level complex types Which presented an interesting problem. There is a need to be able to add new transaction types embedded within the choice statement and there is the need to add new elements outside of the choice statement.

The first pass at solving this problem involved adding an extension point both within the choice statement and at the end of the surrounding complex type. This would allow extensions at both levels. However this resulted in an Unambiguous Attribute Error (UAE), which meant it couldn't differntiate one extension point from another within the same complex type.

To solve this is the "choice within a choice" found in a couple of places within the schema

```
<xs:choice maxOccurs="unbounded">
        <xs:choice>
                <Choice1/>
                <Choice2/>
        </xs:choice>
        <xs:any … >
</xs:choice>
```

This allows one to use Choice1 and still add new elements to the base complex type. It also allows one to add a new Choice of their own need. It does, however, enable some unavoidable unusual consequences. Because the outside choice is repeatable, it allows both Choice1 and Choice2 to be included in one instance document. This would not make good common business sense, so it is a risk. The trade off is to allow proper extensions of realistic business needs for enabling unrealistic business scenarios.

2.5 Enumerations

Traditionally in Retail POS systems, there was the ability to configure the tender in a wide variety of different ways. This was accomplished by using a numeric Tender ID and configuring it to mean different things at each retailer. For example Tender ID =1 could be "cash" at one retailer and "credit card" at another.

This configuration model is a killer for interoperability. To avoid this conflict, ARTS chose to use named enumerations and eliminate the number option. Now only "cash" is used with no ambiguity.

ARTS also recognizes that each vendor has something unique to bring to the table. This means they need the ability to add their own enumerations to the standardized list. ARTS has defined how to do this in the extensibility paper available at NRF.com.

2.6 Domain Drawing Explanation

2.6.1 What is a Domain Drawing?

When ARTS began creating XML Schemas for message integration there wasn't any valid technique for modeling the schema. Data modeling techniques work well for relational information but not for hierarchial information. UML modeling techniques work well for object models but not for relational models (even by stereotyping). The horizontal technique used in many modeling tools is ok as long as the schema is small but falls apart quickly for larger schemas. So ARTS took the best of a variety of techniques and put them together into one that ARTS calls the Domain Drawing.

2.6.2 How do I read a Domain Drawing?

The following Domain Drawing Diagram shows following typographical conventions for the domain drawing diagrams in this document:

- Some data elements can be present in an XML instance document at more than one level. The multiple connections are shown in blue, it is expected that any single XML instance document will make use of only one of the options shown.

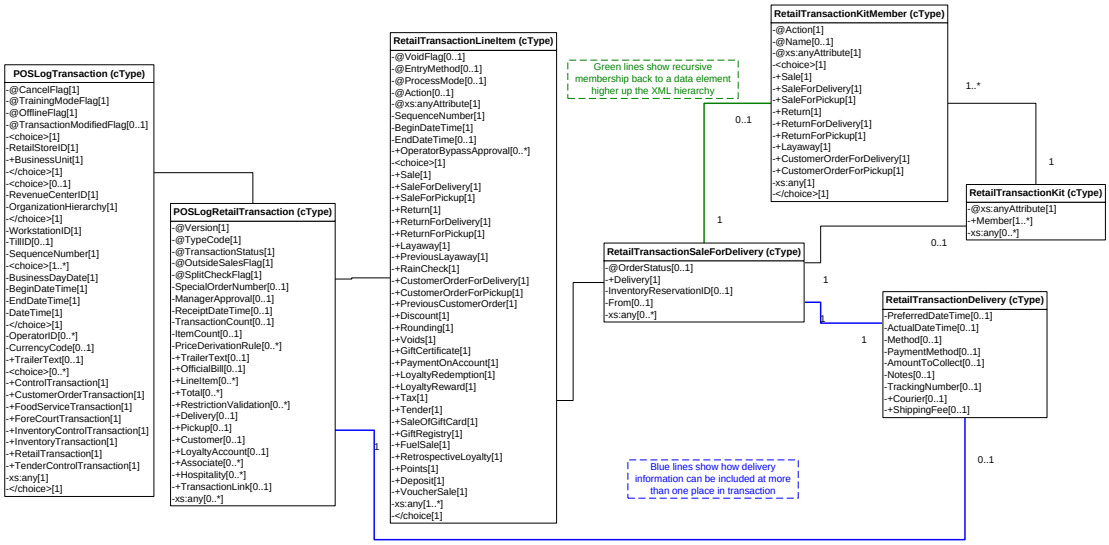


Figure 2 Example Domain Drawing Diagram

To easily understand the relationships contained within the schema, ARTS developed the Schema Domain Model. All other drawing techniques from hierarchical drawings to UML drawings don't properly and clearly display the overall complex type relationships.

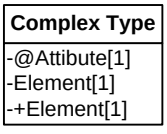


Figure 3: Complex Type Content

The content of each complex type is made up of attributes and elements. The attributes are identified following the XPath nomenclature "@" sign. The other content is the elements. If an element's type definition is a complex type then there is a "+" sign placed in front of the element name.

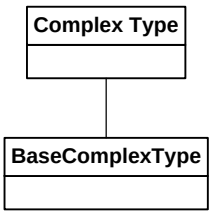


Figure 4: Base Complex Type Model

The Venetian Blind Architecture model is comprised of a number of base complex type building blocks. Elements in one node reference complex types through their type definition.

<xs:element name="asdf" type="BaseComplexType"/>

This relationship is indicated by a straight line between the two complex types.

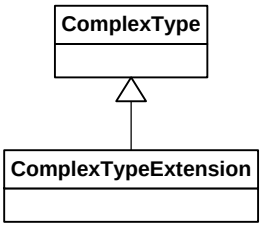


Figure 5: Derivation by Extension

In some cases it makes reuse a complex type in a manner similar to an abstract base class. This is best used when one wants to leverage the existing data and add additional data in the derived class. This technique is called derivation by extension. An example of this is an item. The information in an item is used in several line items such as a sale and a return, each of which can add other information that is only used in that instance, i.e. a return needs a disposition. This is modeled using the line with an arrowhead pointing from the derived class to the abstract complex type.

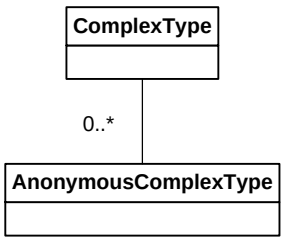


Figure 6: Anonymous Complex Types

The final relationship modeled in the Domain Model is an anonymous complex type. These are characterized by having the node embedded within the higher node.

```
<xs:element name="HigherComplexType" maxOccurs="unbounded">
        <xs:complexType>
                <xs:sequence>
```

```
                    <xs:element name="AnonymousComplexElementType"/>
                    <xs:element name="AnonymousComplexTypeElement2"/>
                </xs:sequence>
            </xs:complexType>
</xs:element>
```

Anonymous Complex Types are modeled with a straight line to the higher node. The cardinality of the anonymous complex type is listed next to the line.

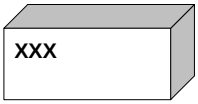


Figure 7: Abstraction for Ease of Explanation

Each of the following discussions focuses on a particular area of the entire schema. As such it requires the duplication of areas already discussed. To simplify the discussion the repeated areas are identified with the above abstraction with a reference to the section where the abstraction is discussed.

2.7 ARTS Common Header

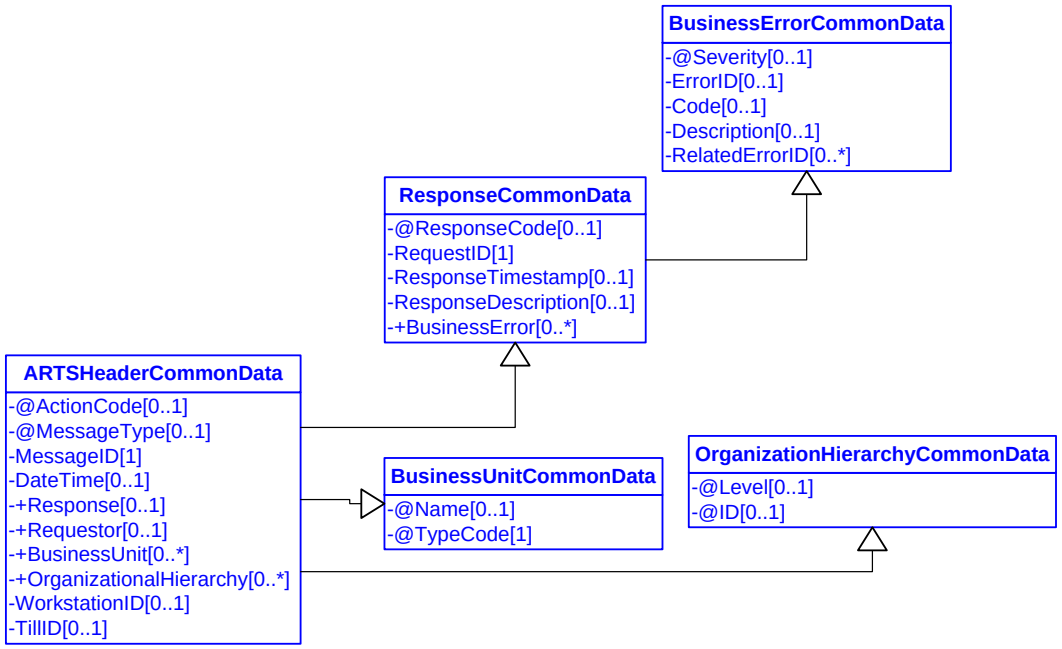


Figure 8: ARTS Common Header Domain View

The ARTS common header is used in all service name schemas. It provides the ability to set session level information and return business error information in one standard format to all SOA implementations.

2.8 How to read an Example XML Instance Document

The following Example XML Instance Document shows the typographical conventions for the Example XML Instance Document in this document:

- The XML fragments are a complete XML instance document showing all schema and name space declarations.
- The XML elements in **red** are interesting features of this particular XML fragment.

```
<?xml version="1.0" encoding="UTF-8"?>
<!—- UseCase: Item Purchase via WWW                                    -->
<!—- Note: OperatorID is missing for WWW Transaction                   -->
<POSLog
        xmlns="http://www.nrf-arts.org/IXRetail/namespace"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.nrf-arts.org/IXRetail/namespace/
POSLogRetailTransaction.xsd"
        Version="1.0">
  <RetailStoreID>WWW</RetailStoreID>
 <WorkstationID>Server2</WorkstationID>
  <BusinessDayDate>2001-09-11</BusinessDayDate>
  <SequenceNumber>8876</SequenceNumber>
  <Transaction>
   <RetailTransaction">
       <LineItem>
           <SequenceNumber>1</SequenceNumber>
           <Sale>
               <POSItemID IDType="GTIN">01234567890123</ItemID>
               <Quantity>3</Quantity>
               <ActualUnitPrice>1.63</ActualUnitPrice>
               <ExtendedAmount>4.89</ExtendedAmount>
               <Delivery>
                   <Address>
                       <AddressLine>325 7th St. NW.</AddressLine>
                       <AddressLine>Suite 1100</AddressLine>
                       <City>Washington</City>
                       <State>D.C.</State>
                       <Country>United States of America</Country>
                   </Address>
                   <PreferredDate>2001-09-12</PreferredDate>
                   <PreferredTime>15:00</PreferredTime>
                   <Method>Courier</Method>
               </Delivery>
           </Sale>
  </LineItem> </RetailTransaction> </Transaction> </POSLog>
```

3. DESIGN DECISIONS

A number of design decisions were made while developing the POSLog Schema; the following sections attempt to capture the thought processes that lead to the construction of the Schema.

3.1 POSLog

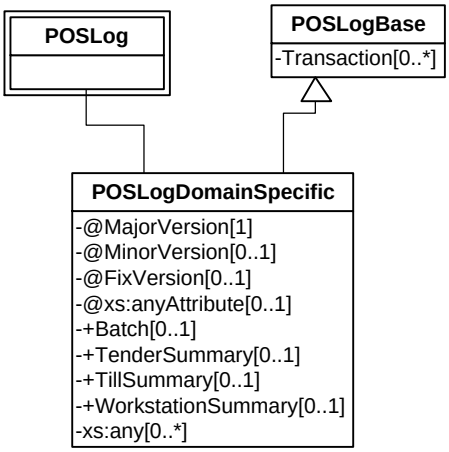


Figure 9: POSLog Root

XML schemas are a hierarchical model. POSLog root allows a variety of different transaction types to be wrapped up under one document. For example, it allows the traditional end of day batch upload by packing multiple transactions together and sending to corporate in one document. Normally this document is sent above store overnight. In the past this was the normal mode of operations for many retailers. However, retailers need access to sales information as soon as it becomes available for such things as merchandising. This is particularly true around Christmas time when there is a need to make sure replenishment and stock movement is optimized for sales. To accomplish this transactions are trickle-fed in near-real time to above-store systems.

3.2 POSLog Transactions

To support various retail processes, POSLog wraps a variety of different transactions under one umbrella. The complete version of POSLog supports the following transaction types:

3.2.1 Order Transaction

An **Order Transaction** supports building a shopping basket. To easiest way to understand where an Order Transaction occurs is during shopping on a website. Here an unknown consumer puts items in their shopping basket. If they decide to buy them, the transaction moves to the next transaction type, a Customer Order Transaction

3.2.2 Customer Order Transaction

A **Customer Order Transaction** supports multi-step transactions normally found in a Quick Service Restaurant (QSR) environment, call center, web, or mobile ordering. For example a customer enters the drive-thru lane and approaches the menu board. There the items in their order are entered and the transaction is stored. The customer drives to the payment window, the order is recalled and they pay for it. They then drive to the pickup window where their food is delivered and they drive off. Each of these steps are

parts of a customer order transaction. When the customer's food is delivered, the Customer Order Transaction becomes a Retail Transaction.

ARTS has defined that a Customer Order Transaction becomes a Retail Transaction when it is tendered and delivered.

3.2.3 Retail Transaction

A **Retail Transaction** is the true heart of Retailing. It reports everything related to selling and returning items. It is conceptually what appears on the customer's receipt plus. It is the customer facing component of POSLog.

3.2.4 Tender Control Transaction

A **Tender Control Transaction** is used to report non-transactional financial data. It supports controlling the internal movement of money and is the store facing tender component of POSLog. To support this, it covers tender actions that can be performed at the POS and in the Cash-Office. E.g. Tender Float, Loans, Pickups, Pay Ins, Disbursements, Safe Drops etc…

3.2.5 Control Transaction

A **Control Transaction** can fundamentally be thought of as an event. Its primary purpose is to feed in-store systems such as Loss Prevention or Store Operations with real-time information. It enables quicker responses to issues as soon as they arise.

It encompasses all administrative and control actions that can be performed at the POS. E.g. Sign On, Sign Off, Lock Workstation, Unlock Workstation, Open Cash Drawer, Session or Shift Close etc…

3.2.6 Foodservice Control Transaction

A **Foodservice Transaction** represents additional foodservice specific events.

3.2.7 Forecourt Transaction

A **Forecourt Transaction** (gas/petrol filling station, convenience store) supports entering pump tests, fuel deliveries, and drive-offs.

3.2.8 Inventory Control Transaction

An **Inventory Control Transaction** allows smaller retailers to leverage the hardware in their store to perform multiple different functions, for example, reporting inventory status. The inventory control component of POSLog allows the use of the POS as an interface to the inventory system for recording goods received, inventory counts, RTV's, etc.

3.3 Customer Order Transaction, Retail Transaction

Basically Customer Orders are interim steps towards delivery of the item(s) **and** collection of the tender, at which time the Customer Order becomes a Retail Transaction.

A Customer Order becomes a Retail Transaction when two criteria are met:

1. The items are delivered to the customer
2. The Transaction is tendered.

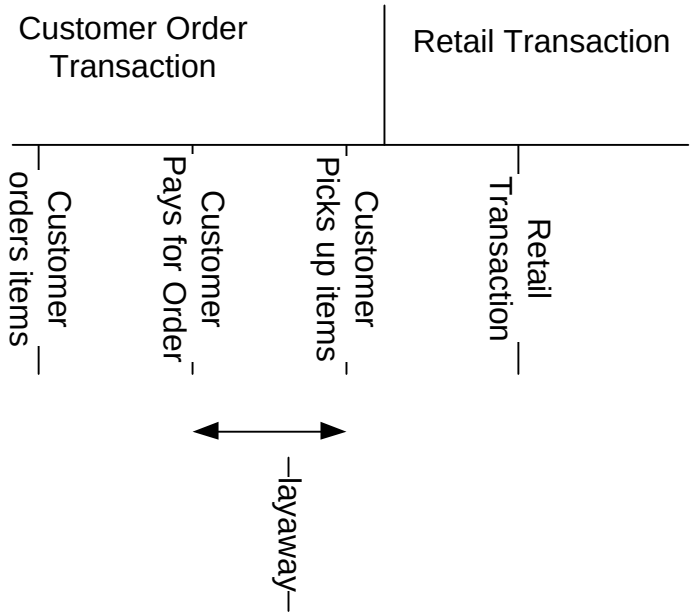


Figure 10: Customer Order - Retail Transaction Timing Diagram

3.4 Payment on Account vs Tender Control Transactions

To clarify when one type of transaction is used over another, the description of Retail Transaction vs a Tender Control Transactions was updated. A Retail Transaction is about the *customer facing* component basically the retail component of the process. A Tender Control Transaction is about the retailer *internal* control of the movement of money.

Why is this important? In normal operations there are non-sales activities which involve money. To interoperate the type of message created by each needs to be clearly understood. For example, a customer comes in to pay their electric bill. This is not a normal sales activity but it is a customer facing activity and therefore it is handled as a type of Retail Transaction Payment-on-Account. Whereas making change by taking money out of the safe to put in the till is an internal operation and falls under the Tender Control Transaction as a Tender Exchange.

3.5 Generally Accepted Accounting Practices (GAAP)

For interoperability needs, ARTS XML has chosen to default to GAAP and make all amounts positive with the use, either adding money to the till or taking money out of the till, being determined in context of the particular instance document.

Basically all money is real, only direction of it movement is different. Either money goes from the customer into the till (sale) or from the till to the customer (return). By this philosophy there is no such concept as negative money and therefore all amounts in the XML are positive.

The data model is not constrained by the interoperability needs of the XML and is more generic in that it has to support more than one philosophy. To that end it allows money to be both positive and negative.

3.6 Transaction Identification, Ticket ID, Receipt Number

The intent of the data model is to allow each Cash Register, POS or other workstation to assign sequence numbers to transactions independently of all other systems, and may re-start numbering transactions from zero every day. If a POS restarts the set of sequence numbers from zero more than once per day then extra effort will be required to provide a unique identifier for each transaction.

Traditionally the retail enterprise built a unique identifier by combining:

- RetailStoreID
- WorkstationID
- SequenceNumber
- BusinessDayDate

Today one can create unique identifiers by using a Universally Unique Identifier. In this case the Transaction “Sequence Number” can be stand alone.

The following are used to support individual transaction numbering:

Transaction ID – primary key in DM

A universally unique identifier (UUID) used to uniquely record a Transaction in the ARTS Data Model. It can be generated in the database or external to the database and stored in the database. This is used as the primary key in the Data Model. When one extracts the information for populating a data warehouse, this is the way to link that information back to the ARTS Data Model.

Transaction “Sequence Number” – generated by the POS

This is the traditional Transaction Sequence Number generated by the POS to identify this particular transaction. It can be unique of itself or in combination with the Workstation ID, Business Day DateTime and Business Unit ID.

Receipt Number - Generated by the POS

Receipt Number can be provided to the customer in order to let them know when their order is ready (fast food). It is typically written on the receipt. It can be the same as the Transaction “Sequence Number”.

3.7 Business Unit and Retail Store ID

Retail Store ID has been abstracted to support any place within an enterprise from the warehouse to the administration center to the retail store. To that end the term Business Unit was defined.

In collaboration with GS1, a Business Unit now relates to any location within an enterprise including the above mentioned distribution center, administration center, call center, web site, distributor, Filler, Transit Cellar, Producer and Grower.

3.8 Business Day Date

The Business Day Date identifies the hours when most businesses in a given locality are in operation or open for business. The conventional business day is from 8 A.M. until 5 P.M. Yet individual businesses may have hours that differ from others or a business may choose staggered schedules.

In a 24-hour operation, a business day typically commences at some early morning time and continues through until roughly the same time the following day. (E.g. the business day of 12-April may commence at 2:00 am on 12th April, and continue through until 1:59 am on 13th April). The purpose of this is to allow the retailer to move 'day-end' activities to a slow period typically early in the morning.

3.9 Default Currency

It is intended that most transactions will express all monetary amounts in an assumed default currency that is agreed between sender & receiver.

In some circumstances, a single stream or batch of transactions may include transactions that express monetary amounts in differing currencies; in these cases the optional element <CurrencyCode> in the transaction header shall denote the default currency for that single transaction.

Note: The POSLog Schema defines any currency amount that can reasonably be expressed in a currency other than the default currency to have an Attribute CurrencyCode that overrides the transaction (or batch) default currency. This allows for a split tender scenario where both the default and alternate tender are applied to the same transaction.

3.10 POSIdentity vs ItemID

In the early days of POS, the number of SKU's was typically limited to a maximum of three digits. This meant that a retailer couldn't uniquely identify every item in their store with its own number. So they added a qualifier to the SKU to allow them to qualify the SKU. For example a retailer might give all long sleeve dress shirts a SKU of 100, and then manage the different colors or sizes with qualifications. ARTS calls this combination a POSIdentity.

With the rise of UPC codes this limitation has effectively gone away and each item is uniquely identified. For most modern systems the ItemID is sufficient to identify the various items in the store.

But even now the UPC system is running into limitations and GS1 has created the Global Trade Identification Number (GTIN) to overcome this obstacle. GTIN's will be slowly replacing the UPC code as the unique item identifier.

3.11 SequenceNumber Elements

The following repeating elements within the context of a RetailTransaction are defined with a mandatory SequenceNumber element:

- LineItem
- OperatorBypassApproval
- RetailPriceModifier
- Tax

While it is arguable that the order of such a repeating item in an XML instance document implies a sequence number, the POSLog Schema includes mandatory SequenceNumber elements for the following reasons:

- Many line items refer to other line items in the same transaction, e.g. A Voids line item voids another line item; Sale & Return of deposit items may also be linked to another line item. It was felt that merely including an element <ItemLink> linking to an implied line item sequence number was insufficient. An implied line item sequence number, does not clear up ambiguities like “are line items numbered from zero or one?”
- All of the repeating items that are defined will at some stage are saved in a database. Normal data integrity rules, require such items to be assigned an identifying sequence number, it is felt that placing an explicit SequenceNnumber element on such data elements at source is a safer practice.
- Inclusion of explicit SequenceNumber elements, allows fraud-detection and other audit systems to verify that batches of Retail Transactions are being sent properly without holes or duplications in the SequenceNumbers.

3.11.1 RTI LineItem/SequenceNumber

Retail Transaction Interface (RTI) is a service-oriented POSLog architecture.

RTI is stateful, so communication is a multi-step conversation and each message contains only new information. RTI is also initiated by the client, which means it has no idea what the current Sequence Number is for each line item.

The communication from the client to the server always starts with a line item sequence number of 1. The communication back from the server to the client is the correct sequence number for this line item.

If an item has some sort of problem, then the server responds by using the line item sequence numbers it has assigned to communicate the problem.

In a shared shopping basket environment, the basket has to keep the sequence numbers as it is possible to communicate this to different servers.

3.12 Line Items

The line items involved in recording the actual sale or return of merchandise come in three major flavors, with each flavor having potentially another three sub-flavors. The complete set is currently:

The Schema defines these line items as explicit data elements, rather than as an abstract Sale line item with a set of attributes denoting the applicable flavor because:

- The set of possible attributes gets confusing as each variation is added.
- Each line item has extra data elements that may be required for that line item and are optional or not required in others. e.g. Delivery or Pickup elements: It is not possible to make such elements required, based on a particular value or values in a set of attributes.

3.13 Kit and Recursion (Combo and Recursion)

The POSLog Schema includes a mechanism for recording omissions, additions and substitutions that are made to a Kit / Combo / Collection / Combination Item when such is being sold. A Kit / Combo / Collection Combination item is a collection of items that is priced & sold as a single item, but which is exploded into its constituent items for the purposes of tracking inventory.

In Retail, it is known as a “Kit”. In foodservice it is known as a “Combo”. Both elements have been added for credibility in their respective domains.

In some cases the items in a kit can be purchased separately. If each item is scanned separately at the POS, then when all the items which belong to a kit have been scanned, a “best pricing algorigthm” kicks in and provides the discount associated with the kit.

The data element <Kit> was defined to have a list of <Member> elements which come in the same Sale, Return and Back-Order flavors previously discussed. Each member has an Action attribute that denotes how the kit member affects the kit. Possible values of the Action attribute are:

- **IsPartOf:** The item is a pre-defined member of the kit, and has no effect on the price of the kit. Many implementations may not explicitly include these kit members in a transaction, although if the kit member has special delivery or pickup instructions then that would be a reason for including these kit members.
- **AddsTo**: The item is not defined to be a member of the kit, but is added to the kit for this transaction. The <Quantity>, <UnitPrice> & <ExtendedAmount> elements record how the inclusion of this kit member affects the price of the kit.
- **IsRemovedFrom**: The item is normally part of the kit, but is removed from the kit for this transaction. The <Quantity>, <UnitPrice> & <ExtendedAmount> elements record how the removal of this kit member affects the price of the kit.

3.13.1 Kit Recursion

The recursive definition of a Kit and its members allows complex taxation, delivery & pickup and price modification rules to be applied to individual members of a kit for a particular transaction. e.g. A stereo is a kit made up of Amplifier/Tuner, CD, Tape Deck and 2 sets of speakers, but the particular speakers are out of stock, and can be picked up by the customer from another store across town.

3.14 Location of Pickup or Delivery Data

When an item being sold or returned has to be picked up or delivered, extra data about pickup and delivery address, date & time has to be recorded in the transaction. The schema allows such information to be included in the transaction in two places:

- **Transaction Header:** All items sold for delivery in the transaction are to be delivered to the same address, at the same time. The data is only recorded once

in the transaction header. The existence of SaleForDelivery, or BackOrderForDelivery flavored line items cannot make the inclusion of the delivery data at the transaction header mandatory.

- **Line Item Items:** Each item sold for delivery in the transaction can be delivered to a different address, at a different time. When all items are to be delivered to the same address at the same time, the data is repeated in each line item. The delivery data can be made mandatory at the line item level.

Although the POSLog Schema allows Pickup & Delivery data to be recorded at both locations in an XML Instance document, it is expected that a particular implementation will make use of only one, depending on the policies of the business concerned.

3.15 Discounts or Retail Price Modifier

When an item is sold for a price that is different from the regular lookup price, the price modification of that item may be recorded in one or more RetailPriceModifier elements. The RetailPriceModifier element records:

- **Percentage Discounts & Surcharges:** Add or subtract a percentage of the unit price to the item.
- **Monetary Amount Discounts & Surcharges:** Add or subtract a monetary amount to the item.
- **Monetary Amount Replacement:** No discount or surcharge, simply here's a new price.

The RetailPriceModifer element includes a SequenceNumber, to record the order in which multiple retail price modifications were performed on the unit price of a particular line item.

In the normal sequence of events, once all price modifications have been applied to the unit price, the taxation rules are applied to calculate the amount of tax to be collected. The tax collected may then be reported per line item, per transaction, or both.

A common historical practice is to apply discounts at the transaction level. When such transaction level discounts are applied across the entire transaction, two assumptions are being made.

- The discount is applied equally across all items in the transaction.
- All items in the transaction are taxed at exactly the same rate.

Unfortunately in many instances neither of these assumptions stands true. In modern systems, transaction level discounts are in fact applied to particular line items based on complex rules set by the business. Examples of such complex rules include: Highest Price Item, Highest Margin Item, Highest & Lowest Taxed Items, etc.

In POSLog V6 another type of discount has been added. This discount is reported as a line item similar to the transaction level discount but it has links to the line items for which it applys. This allows line item 1, 3, 6 to qualify for a discount related to all three items being bought similar to the kit type discount. This is not to be confused to a kit as a kit is not a discount but the price for that collection of items.

The POSLog Schema includes both a <Discount> element at the Line Item level that is used for recording transaction level and multi-line item related discounts, and a <RetailPriceModifer> element in the <Item> complex-type that is used for recording specific item level discounts and price modifications, to allow maximum flexibility.

Because of the difficulties of having Back-Office and corporate systems reproduce the exact discount & taxation calculations that were performed by the POS, the POSLog Schema includes elements that allow such calculations to be performed once and once only by the POS, and then reported to and used by all other store systems.

3.16 Tender IDs

A long time ago, it was common practice to have a numeric Tender ID (1,2,3,) to define different tender types. In part due to the limitations of the hardware and communication protocols. Interoperability suffers greatly with this approach. ARTS decided that from an interoperability perspective TenderID's (although common in the industry) were bad. That's because both sides need to have the same mapping and that's a problem especially in a SOA world. In the XML world enumerations help alievate this problem by providing an English equivalent for each tender type, "Cash", CreditDebit", etc.

Some times one wants to further identify the tender information, for example, Visa, Master Card, are all types of CreditDebit. To solve this ARTS put in place a concept called a SubTenderType. If additional sub-types are neede, i.e. the euro, then this enumeration can extend this following the ARTS extension method.

3.17 POS Transaction State Diagram

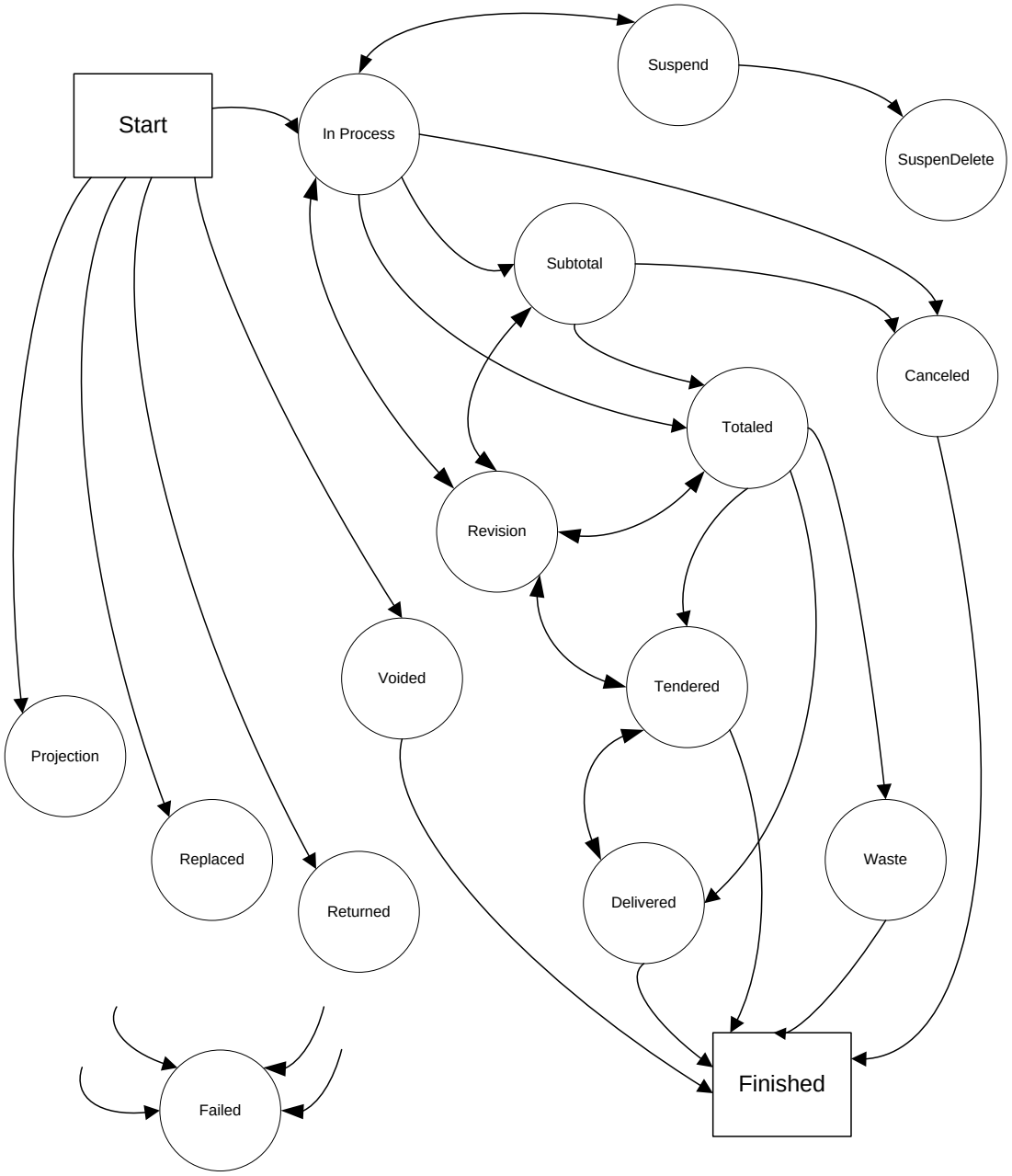


Figure 11: POS Transaction State Diagram

Transaction status identifies the various states in which a transaction can exist.

- **Cancel** – Cancel is used to designate a transaction line or entire transaction that was reversed PRIOR to the finishing of a transaction but after items were sent to the kitchen.
- **Delivered** – A transaction that has ownership of the listed items transferred to the customer
- **Failed** – A transaction that has problems forcing non-completion.

- **Finished** – A completed transaction, i.e tendered and delivered.
- **InProcess** – Items entered during a one-behind or on-the-fly order entry mode.
- **Projection** – A projection of the items needed for a given time period. Used in analyzing the needs, both manpower and inventory, for the store.
- **Replaced** – This transaction replaced another transaction. Typically used to correct a problem with the original transaction.
- **Returned** – Indicates this is transaction is returned.
- **Revision** – This transaction is a revision of the original transaction.
- **Subtotal** – Indicates an intermediate total is calculated prior to finishing.
- **Suspended** – A transaction that has been stored to be recalled at a later point by the same or different POS during a customer order. A suspended transaction must be recalled to be changed.
- **SuspendDelete** – Deletes a transaction after it has been stored. This has potential inventory implications because food preparations may have begun.
- **Tendered** – Indicated the transaction was tendered.
- **Totaled** – The final total prior to tendering.
- **Void** – Designates a transaction line or entire transaction that was reversed as part of a finalized retail transaction, i.e. AFTER tendered and delivered.
- **Waste** – reports this transaction as waste.

3.18 Order Item Status

Individual items on a customer order can be in various stats prior to completion of the order.

- **Hold** – The item has been entered but don't start preparation.
- **Dispatch** – Prepare the item now.
- **Sent** – The item has previously been dispatched to the kitchen and prepared.
- **Delivered** – The item has been delivered to the customer.
- **Cancel** – The item was cancelled after being entered and dispatched to be prepared.
- **Void** – The item was voided after the retail transaction was created.

3.19 Transaction Totals

NOTE: Positive GAAP amount is used when dealing with tender. But totals are not money but numbers. Therefore it is possible to have negative totals.

3.19.1 Gross, Net and Grand

A question that regularly comes up is how to report all the different totals and maintain their relationships. This has many uses from what gets actually printed on the receipt to being able to report it from a business intelligence point of view.

There are two major groupings depending on taxes. The first is the Sales Tax type where tax is **ADDED** to the price of the goods at the point of sale.

- **TransactionGrossAmount** is the total excluding discounts and surcharges without taxes
- **TransactionNetAmount** is the total including discounts and surcharges without taxes
    - TransactionNetAmount = 165.00
- **TransactionTaxAmount** is the total excluded taxes, i.e. sales tax, on the transaction
    - TransactionTaxAmount = 41.25
- **TransactionGrandAmount** = TransactionNetAmount + Taxes
    - 206.25 = 165.00 + 41.25

The second grouping is the VAT Tax type where tax is **INCLUDED** in the price of the goods at the point of sale. This is because no one ever sees it. It can be reported on the bottom of the receipt but it is basically included in the price shown to the customer.

- **TransactionGrossAmount** is the total excluding discounts and surcharges
- **TransactionNetAmount** is the total including discounts and surcharges
    - TransactionNetAmount = 206.25
- **TransactionTaxIncludedAmount** is the total amount of the tax included in the price of the items.
    - TransactionTaxIncludedAmount = 41.25
- **TransactionGrandAmount** = TransactionNetAmount
    - 206.25 = 206.25

Another way to look at it, TransactionGrandAmount is the amount the customer pays.

There can also be a Gross Amount:

Gross Amount = Net Amount + Discounts – Surcharges. So you could have a 10.00 euro item with a 10% discount and included VAT.

Gross Amount = 10.00

Net Amount = 9.00

Grand Amount = 9.00 (amount the customer pays)

3.19.2 Other Transaction Totals

- **TransactionNonSalesAmount** - These are charges during a transaction that are not related to regular sales. Can be things like pay ins, pay outs, rental fees, stamps, ... Depends on how the retailer wants to account for these items
- **TransactionPurchaseQuantity** - This is the number of items in the transaction. A line is commonly found on the bottom of receipts, "number of items purchased = X"

  - Does this include: voided items, returned items, packages (for example if we sell tomatoes in a package)? What is the business around this counter: for example to print number of items on receipts so the supervisor can count the items to verify it's the correct amount? Yes if it's a package of tomatoes, it would generally be a count increase of "1" per scannable item. How it is used is a retailer decision. It will depend on how they measure things. ARTS has a KPI effort to help retailers understand some options around evaluating their business.

- **TransactionSubtotal** – On many receipts a sub total is written on a separate line. This is the total sometimes before discounts, at other times it is before taxes are applied. It is typically the sum of the individual line items on the receipt.
- **TransactionFoodstampTotalAmount** - it is perfectly possible to comingle foodstamp items and non-foodstamp items on the same receipt. This is the total of all the foodstamp line items, sometimes reported separately on the receipt.
- **TransactionCouponTotal** – the total amount given for all coupons used on this receipt
- **TransactionCouponCount** – the total number of coupons used on this receipt
- **TransactionTotalSavings** – this is the total of all savings (loyalty, coupon, etc) given on this receipt.
- **TransactionTenderApplied** – identifies the type of tender used in payment on this receipt.

4. TAXES

4.1 Tax Overview

Basically there are two types of Taxes: Sales Tax and VAT (Value Added Tax). Sales taxes are those that are added to the price at the time of purchase. VAT taxes are already included in the price of the item. Governments are ingenius in their creation of new taxes and fees. To support all these various flavors ARTS has included an extensible attribute called “TaxType”.

There are also situations where the taxes are forgiven or exempted.

In addition, there are two points in the transaction where taxes may be applied or not, either to the whole transaction amount, and/or to the individual line items.

4.1.1 Tax Exempt

The monetary value of the retail transaction line that would normally be taxable but, because of special circumstances, is not. For example, the sale of machinery is normally subject to sales tax. However, when the machinery is used in manufacturing, some taxing jurisdictions consider the sale exempt. A certificate is usually required for an exemption to apply

Diplomats are tax exempt, while a diplomatic aide may receive a tax reduction of 5%.

There can be tax exempt - people who have a tax exempt status

a. Senior Citizen
b. Foreign National
c. Non-profit organization
d. Duty Free – import or export

4.1.2 Tax Forgiven (Tax Override)

Tax Forgiven is very similar to Tax Exempt but does not require the certificate. It is implicit in the products bought and how they are paid. This typically covers things like items bought with foodstamps or WIC (Women, Infant, Children).

With foodstamps, the government has developed a list of products which get this treatment and in some cases entered into contracts with particular brands. When a customer purchases an item with their foodstamp, the POS validates the items against the government approved list then does not charge taxes against these items.

Common Tax Override scenario is Return Item to a Store in the same chain but located in a different Tax Authority (different Rate). POS will allocate Tax assuming local Tax Rule, Operator has to override to match Receipt and correctly Refund customer

4.2 Tax Discussion

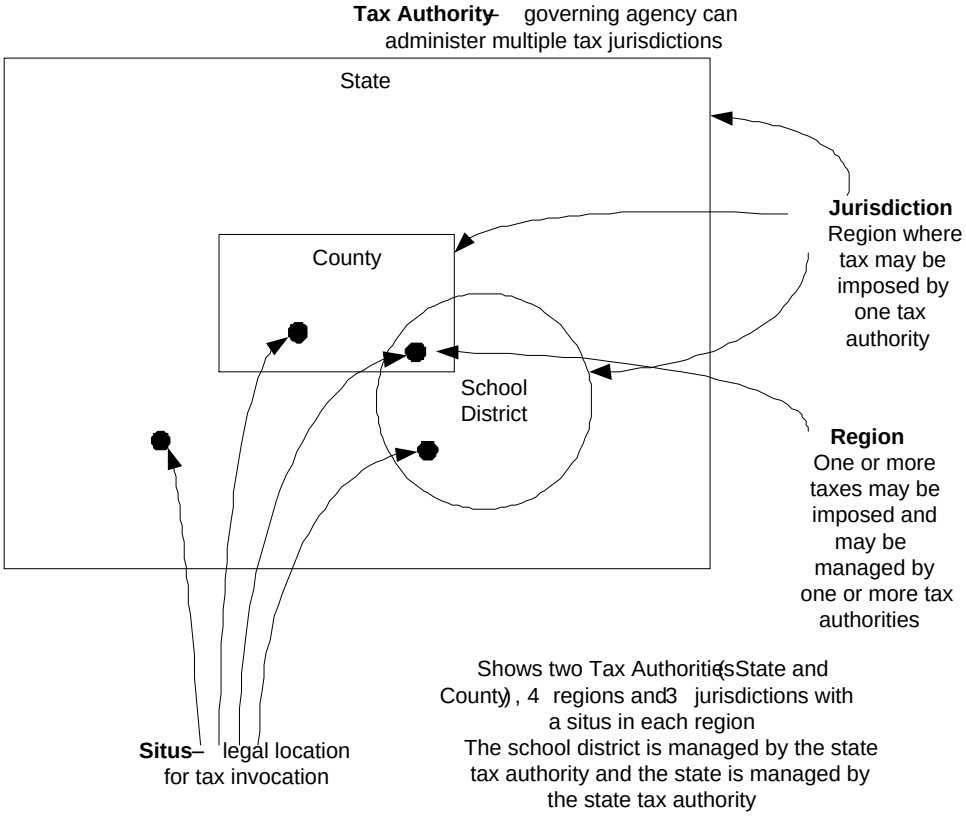


Figure 12: Tax Relationship Overview

In the United States, each state has its own list of taxes with their own names. While there may be overlap of some types, each State has at least one or two that are unique to themselves. There is no way to explicitly enumerate this enormous and dynamic list of taxes in the POSLog schema. Therefore, POSLog relies on a generic classification for taxes (VAT, Sales) with specific instances of taxes being identified as TaxRuleID. When used in conjunction with the TaxAuthority and TaxJurisdictionID, one can derive a unique meaning for the tax. Always assuming these values follows the definition created by the Tax Authority government agency.

5. RETAIL TRANSACTION INTERFACE (RTI)

5.1 Overview

The ARTS XML Retail Transaction Interface (RTI) is a comprehensive set of XML message specifications for creating, populating, and completing retail transactions that reduces the time and cost of integration to retail transaction functions. The messages are fully in alignment with the ARTS SOA Best Practices Technical Report.

RTI and the service behind it, provided by the POS application, together provide a SOA Composite Service. RTI with the service behind it is consistent with the Retail Transaction Service identified by the ARTS SOA Blueprint for Retail, and the interface definition follows the SOA Service Interface definition recommendations laid out by the ARTS SOA Best Practices for Retail.

Use of this standard enables retailers to leverage POS retail transaction business logic, now also available as a Service, for multiple user touch points such as kiosks, self-checkout, and handhelds. The RTI is consistent with the ARTS XML common data library, with the business uses cases specified in the Use Case Summary section of the ARTS XML Retail Transaction Interface Charter, and the ARTS XML POSLog schema.

The figure below depicts the RTI being used with a POS system to support multiple touch points.

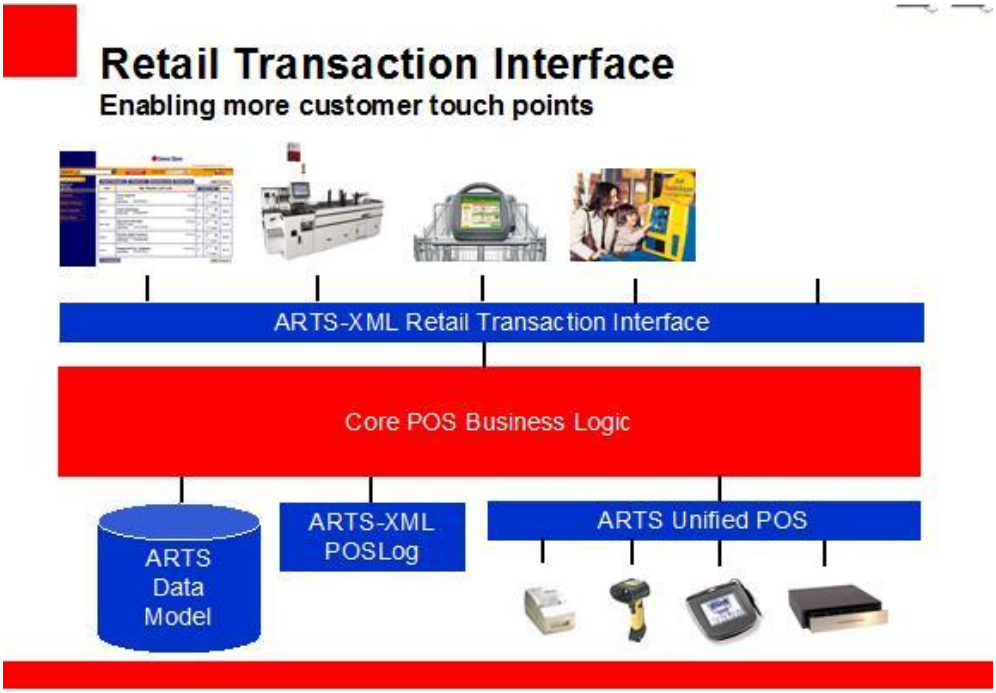


Figure 13: RTI Overview

RTI is only one service interface that an application may employ for a customer touch point solution. Other services that re-use existing POS logic may also be needed to complete the solution as illustrated in the figure below. In this example, a self check out touch point is depicted and in addition to selling items, the self checkout kiosk will need tender control – pickups, etc…, as would a regular check out lane. This functionality

would be handled through a tender control service interfacing with the existing POS application. The self checkout solution would also need a service, such as WSPOS, to interact with devices in order to handle such requests as a “printer out of paper” message.

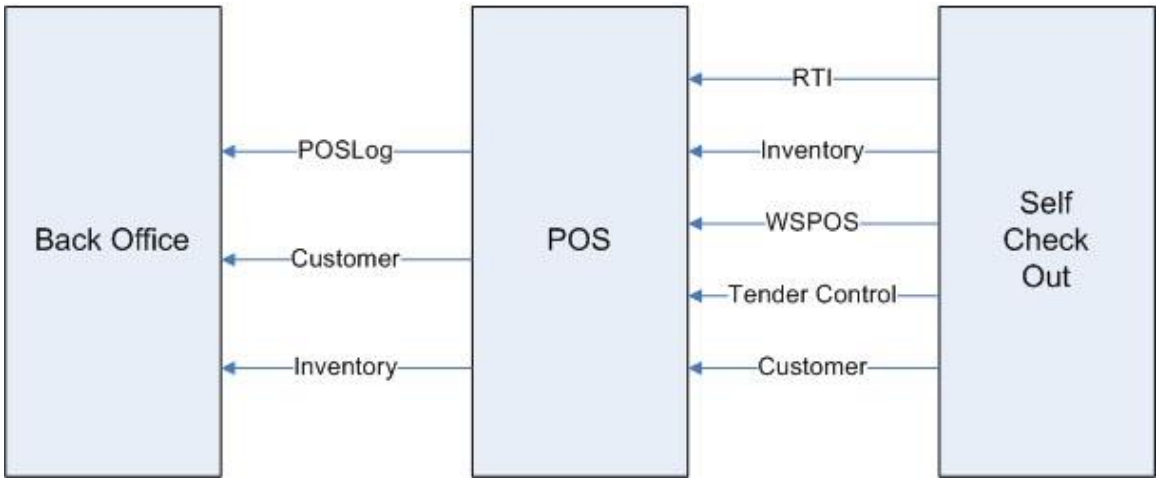
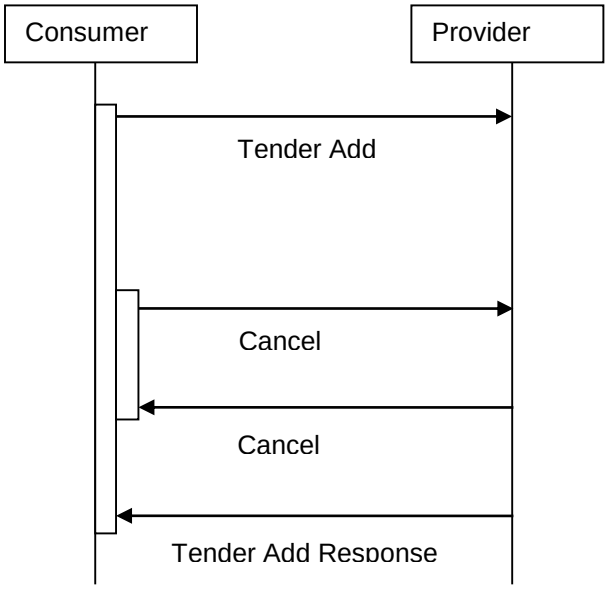


Figure 14: RTI Usage Scenario

5.2 RTI Usage Specifics

For version 1 of RTI, all messages are **synchronous** request and response pairs. There are no use cases dealing with asynchronous events. Nonetheless, the service consumer needs a means to interrupt requests whose execution takes too long on the service provider's end. Or, the user might want to interrupt a previously-issued request that won’t complete.

To enable such use, a **second synchronization group** was introduced. In the first version of RTI it contains exactly one request method: cancel. The use of cancel, again, is strictly synchronous: once the service consumer requests cancel, the service provider must respond to that request and then respond to the preceding request with an appropriate return value. More specifically, the response to both cancel and the preceding request should be issued immediately. If there is no preceding unanswered request, of course, only respond to the cancel.



A service implementation behind RTI is **stateful**. It means that the service keeps client interaction context between client requests. This client interaction context is often referred to as the service or application state. A stateful service usually implies a multi-step conversation (also called a session) since there would be no reason to preserve the state for single step interactions.

It is often the case that not all client requests make sense at all stages of the conversation. Moreover, there is a certain business flow that describes the behavior of the service as the conversation moves from one stage to another. To model such systems, designers often use a concept of a state machine.

It is important to note that the state of the state machine is different from the application state discussed above. The concept of the state machine state describes a certain stage of the business process implemented by an application. The concept of the application state describes the whole context (values of all the significant variables) of the application. Thus, the state of the state machine is also a part of the application state.

Each state determines what inputs (client requests) can be processed at a particular stage of the conversation. When a valid request is received, it is processed by the service which may also trigger a change of state (also called a transition). After the transition is complete, the service is ready to accept new client requests that are valid in the new state. Usually a service can process only one request from the same client at a time.

Different implementations of the service behind RTI would have different state diagrams. In other words, a state diagram describes a particular service implementation and as such cannot be included in this specification. This means that RTI imposes no limitation of which particular requests can be sent to the service at any particular stage. If a method is invoked at a stage of the conversation where it is not valid, the service should return the illegal operation error.

In the example XML, "**Customer Order**" is used here in RTI since it comes from the ARTS data model and we want to be consistent with that. The term was introduced into POSLog V2.2.

6. NEAR USAGE

ARTS created the Notification Event Architecture for Retail to manage sending events to help managers run their organization. There are numerous examples of how to send events during a transaction.

7. DOCUMENT HISTORY

Version History

Ver	Date	Sections	Description of Change
1.0	2002-03-31	All	Candidate Recommendation
2.0	2003-04-02	All	Added Tendering Added Foodservice Example Created Volume 2 – Tendering Created Volume 3 – Taxation
2.1	2004-02-10	All	Bug Fix Version Updated Hierarchy Diagrams Updated instance documents for use in Conformance Program
2.2	2005-02-21	All	Reduced the number of mandatory elements for all 2.1 use cases Added print dispatch docket use case The POSLogTotals type in POSLogLibrary.xsd has a Reason element that is of type of string. changed to use type POSLogReason like the rest of the schema Complex type POSLogAmount is defined allowing both positive and negative values. This allows different systems to use different rules on how the signs are defined. This will break the interoperability. Defined POSLogAmount to allow only non-negative (greater or equal to zero) decimal values. NOTE: this is now a common data complex type. Added Currency Code to Retail Transaction Total. Use the common data monetary amount complex type The ARTS XML definition has an <xs:choice> between BusinessDayDate, BeginDateTime and EndDateTime. The latter option, EndDateTime, has a minOccurs="0" setting. This means that one of the options in the <xs:choice> is to have no date at all. Removed the minOccurs="0" setting POSLog requires an OperatorID. This doesn't work for Unattended Operations. Made POSLogTransaction/OperatorID Optional RetailPriceModifier.PreviousPrice element changed from "xs:decimal" to POSLogUnitPrice cType: RetailTransactionDiscount The ARTS XML definition of this complexType does not include the <xs:any> and <xs:anyAttribute> extension point definitions, which means we are unable to add this attribute and these elements and still produce POSLog v2.1 conformant XML.

			Added these extension points. Add **Element:** Amount, **Element:** PriceDerivationRule, **Attribute:** Enabled to RetailTransactionDiscount cType: RetailTransactionPriceDerivationRule added the <xs:any> and <xs:anyAttribute> extension point definitions, added **Element:** ReasonCode, **Element:** AssignmentBasis, **Element:** Method, **Element:** AccountingType, **Element:** ApplicationType, **Element:** PostProcessType, **Element:** ReferenceID, **Element:** ComparisonBasis, **Attribute:** IncludedInBestDeal, **Attribute:** AdvancedPricingRule to RetailTransactionPriceDerivationRule When ItemID is chosen, there is a problem that the class and/or Dept to are not put in MerchandiseHierarchy. Made following change <xs:choice minOccurs="1" maxOccurs="4"> <xs:element name="POSIdentity"/> <xs:element name="ItemID"/> <xs:element name="OpenDepartment"/> <xs:element name="SpecialOrderNumber"/> </xs:choice> <xs:element name="MerchandiseHierarchy" minOcurrs="0" maxOccurs="unbounded"/> System should be able to understand what type of item lookup code was used at POS. Currently POSIDType is defined simply as string. Abstracted ItemID to common data and created an enumerated list based on UPOS. cType: CustomerOrderForDelivery – added <xs:any> and <xs:anyattribute> Added **Element:** StatusChanged, **Element:** QuantityPicked, **Element:** QuantityShipped, **Attribute: PreviousOrderStatus to Customer Order for Delivery** ProratedFlag attribute inside RetailTransactionDiscount is given a default. Rounding element is made optional. consider that this data is transferred between systems, and the entire hierarchy may not be specified in the foreign system. Made Merchandise Hierarchy repeatable and added an embedded choice around the remaining item identifiers.

			-

8. COMMITTEE MEMBERSHIP

8.1 Version 5.1 (Customer Schema)

Chairman:

David Dorf	Oracle

Members:

Marty Wolfe	IBM
Sam Palanivel	Lowes
Michael Julson	Escalate Retail
Leonid Rubakhin	Epicor
Atul Changela	Xpient Solutions
Joe Finizio	RSPA
Rick Goertzen	S4
Richard Halter	ARTS

Contributors:

David Dorf	Oracle
Marty Wolfe	IBM
Leonid Rubakhin	Epicor
Richard Halter	ARTS

8.2 Version 5.0 (Tax Calculation Schema)

Chairman:

Scott Gamel	ADP Taxware
John Glaubitz (after 9/2008)	Vertex

Members:

Perry Kramer	BJ's Wholesale Club
Laurie Moy	ADP Taxware
Gary Rucinski	ADP Taxware

Contributors:

Richard Halter	ARTS
Stuart McGrigor	ARTS
Jerry Rightmer	Oracle Retail
Jeff Sheldon	Micros/Datavantage
Leo Rubakhin	NSB
Dennis Blankenship	Clicks and Mortar
Paul Dellevigne	Vertex
Don Fuga	Vertex
Kirsten Wright	Retail Anywhere
Angela Seidel	GK Software

Sherman Wilson	Accenture
Dan Conway	Oracle
Jean-Louis Humblot	E-LaSer SA
Rowan Bradley	Herbert Group
Greg Lowes	Waitrose
Hendrik Scheider	Wincor Nixdorf
Johnson Fan	ADP Taxware

8.3 Version 4.0 (Retail Transaction Interface Schemas)

Chair:

Julia Rockwell	IBM

Members:

David Dorf	Oracle
Richard Halter	MIC
Rajaraman Hariharan	IBM
Graham Hill	PCMS
Tim Hood	SAP
Michael Julson	Escalate Retail
Perry Kramer	BJ's Wholesale Clubs
Frank May	Microsoft
Bill Noonan	IBM
Jerry Rightmer	Oracle
Julia Rockwell	IBM
Leonid Rubakhin	The NSB Group
Hendrik Scheider	Wincor-Nixdorf
Angela Seidel	GK SOFTWARE AG
Jeff Sheldon	Micros
Dean Sleeper	AccessVia

Contributors:

Julia Rockwell	IBM
Richard Halter	MIC
Michael Julson	Escalate Retail
William Noonan	IBM
Rajaraman Hariharan	IBM
Leonid Rubakhin	The NSB Group
Hendrick Scheider	Wincor-Nixdorf
Angela Seidel	GK SOFTWARE AG
Viswanath Srikanth	IBM

8.4 Version 3.0.0 (Foodservice)

Chairman:

Moin Moinuddin	Microsoft

Authors:

Autl Changela	Xpient Solutions
John Muhlberger	Par Technologies
Nick Scavanoe	Accuvia
Paul Gay	Epson
Richard Halter	MIC
Bob Bruce	Control Products
Charlie Souhrada	NAFEM
Dave Moorman	PCMS Datafit
Dave Van Horn	SofTechnics
Drew Seale	Xformity
Kazunori Hirano	Toshibatec Japan
Yasuo Sakami	Foresight
Frank May	Microsoft
Graham Hill	PCMS Datafit

8.5 Version 2.2 (POSLog)

Chairman:

Richard Halter	ARTS-MIC

Authors:

Richard Halter	MIC

Reviewers:

S.Hanai	IBM Japan
K.Hirano	TOSHIBA TEC

Contributors:

Jerry Rightmer	360 Commerce
Luis Oliveira	360 Commerce
Glen Tarsha	ADS Retail
Dennis Blankenship	Clicks and Mortar
Paul Fala	CRS Retail Systems
Bob Baker	CRS Retail Systems
Jeff Sheldon	Datavantage
Francisca Vincente-Tamarin	El Corte Engles
Paul Gay	Epson
Seiji Mori	Fujitsu
Ronald Scholz	GS Software AG
Peter Rush	Harrods Limited
Alan Lipso	Hewlett Packard
Reza Attarha	Logware
Maria Borelli	Logware

John Fluke	IBM
Julia Rockwell	IBM
Shisei Hanai	IBM
Yoshitaka Kido	IBM
Frank May	Microsoft
Kazutoshi Ota	Microsoft
Soichi Fuji	Microsoft
Hiroshi Matsumoto	NEC
Stuart McGrigor	NRF-ARTS
Leonid Rubachin	The NSB Group
Dave Moorman	PCMS Datafit
Akira Matsuyoshi	RSI Corporation
Ronald McEvoy	Soft Solutions
Maroun Atallah	Soft Solutions
Tim Hood	Triversity
Dave Parr	Unipower Solutions Europe, Ltd
Kazunan Mandai	Vinculum Japan
Rainer Kramer	Wincor Nixdorf International Gmbh

8.6 Version 2.1 (POSLog)

Chairman:

Richard Halter	ARTS-MIC

Authors:

Richard Halter	ARTS-MIC
Stuart McGrigor	ARTS

Contributors

David Parr	Unipower Solutions
Kazutoshi Ota	Microsoft
Soichi Fuji	Microsoft
Shisei Hanai	IBM
Yoshitaka Kido	IBM
Akira Matsuyoshi	RSI Corporation
Seiji Mori	Fujitsu
Hiroshi Matsumoto	NEC
Kazunan Mandai	Vinculum Japan
Lora Bouchard	Fahrenheit Technology
John Fluke	IBM
Francisca Vincente-Tamarin	El Corte Ingles
Leonid Rubakhin	NSB
Milind Dharkar	Skillnet, Inc.
Amin Sikander	Oracle
John Hervey	PCATS
Scott Kurth	ISS
Frank May	Microsoft

8.7 Version 2.0 (POSLog)

Chairperson:

Richard Halter	ARTS-MIC

Authors:

Richard Halter	ARTS-MIC

Contributors:

Jeannine Ralston	The Limited, Inc.
Perry Kramer	Ames Department Stores, Inc
Ann McCool	Radio Shack Corporation
Tim Reagan	Radio Shack Corporation
John Rohland	Blockbuster, Inc.
Nancy Hudak	Target Corporation
Timothy Hood	Triversity
Jim Galloway	AfterBOT
John Fluke	IBM
Ellen McCubbin	IBM
Michael Maximilian	IBM
Barry Henderson	ISS Retail
Paul Olson	Blue Martini
Yoko Nakagawa	Drummond Group
Mike Dillon	Drummond Group
John Hervey	NACS
Brian Blauvelt	CRS Retail Systems
Jerry Rightmer	360 Commerce
Steve Gannon	360 Commerce
Aaron Link	SIVA Corp
Ed Shirey	ELSWare Technologies, Inc.
Richard Mader	ARTS
Stuart McGrigor	ARTS
Tony Montgomery-Smith	PCMS-Datafit

8.8 Version 1.0 (POSLog)

Chairperson:

Tony Montgomery-Smith	PCMS-Datafit

Authors:

Richard Halter	Apigent Solutions
Stuart McGrigor,	ARTS
Tony Montgomery-Smith	PCMS-Datafit

Contributors:

Alice Cain-Nelson	Nordstrom
Doug Jones	Target
John Carrier	Shell International & IFSF
John Hervey	NACS
Jon Ransdell	McDonald's
Tryggvi Thordarson	HBI
Jim Galloway	AfterBOT

9. DOMAIN GLOSSARY

Term	Definition
Access Enumeration	RTP, Batch, Phone, Web, Kiosk, Administrative
Account	The financial accounting unit for one or more stored value instruments as maintained by the SVA. The account represents the stored value due the holder of the account.
Account ID	The globally unique identifier of the stored value account
Account Type	An attribute type that allows the Stored Value Application to interrogate and execute business rules. Examples include (gift, store credit, merchandise return, layaway, phone, etc.)
Account Balance	The current balance of the account in a given unit of measure
Account Balance UOM	The units in which the account balance is tracked – Enumeration (currency, points, minutes)
Account Balance Locale	Locale code specifying the geopolitical locale determining the account balance
Account State	Active, Inactive, Suspended, Overdrawn
Account Type	GiftCertificate, GiftCard, MerchandiseReturn, StoreCredit, Phone, Layaway, Loyalty
Activation Method	The method used to activate the SVI: RTP, batch, phone, administrative, web
Application Interchange Profile	Indicates the capabilities of the card to support specific functions in the application
Application Identifier (AID)	Identifies the application as described in ISO/IEC 7816-5
Application Transaction Counter (ATC)	Counter maintained by the application in the ICC (incrementing the ATC is managed by the ICC)
Application Usage Control	Indicates issuer's specified restrictions on the geographic usage and services allowed for the application
Application Version Number	Version number assigned by the payment system for the application
Authentication Data	The data used to authenticate the SVI: PIN, control number, CVC, CVV, etc. (optional)
Authorisation Response Code	Code that defines the disposition of a message
BackOrderForDelivery	A flavor of retail transaction line item whereby the customer wanted to purchase an item, but it is unavailable. The store is placing a special order on behalf of the customer, and will deliver the item when it becomes available. The customer may or may not pay for the item at the time of placing the order.
BackOrderForPickup	A flavor of retail transaction line item whereby the customer wanted to purchase an item, but it is unavailable. The store is placing a special order on behalf of the customer, and the customer will pickup the item when it becomes available. The customer may or may not pay for the item at the time of placing the order.

Term	Definition
Base Unit of Measure	Currency code or other UOM code including points, miles, etc
Cancel	The process of Terminating or "canceling during" a transaction before the transaction has been completed and posted. The application may or may not keep an audit trail.
Cardholder Verification Method (CVM) Results	Indicates the results of the last CVM performed
Collection	A collection of items that is priced & sold as a single item, but which is exploded into its constituent items for the purposes of tracking inventory.
Collector	The ID of the party collecting value for the SVI
Combination	A collection of items that is priced & sold as a single item, but which is exploded into its constituent items for the purposes of tracking inventory.
Combo	A collection of items that is priced & sold as a single item, but which is exploded into its constituent items for the purposes of tracking inventory. Typically used in the foodservice industry
Cryptogram Information Data	Indicates the type of cryptogram and the actions to be performed by the terminal
Expiration Date	The date the stored value instrument expires
Holder	The ID of the party that owns an activated instrument.
ICC	Integrated Circuit Card
Instrument	The physical media containing the identifier for a stored value account maintained by the stored value application.
Instrument State	Active, Inactive, Expired, Stolen, Damaged
Issuer	The ID of the party that activated the instrument.
Issuer Application Data	Contains proprietary application data for transmission to the issuer in an online transaction
Item Potent	The process of sending a transaction message out and having a method to verify that the message was received and the necessary response action taken. It guarantees that the transaction was completed and that the "system" did not go down during the transaction time. This process guarantees that only one message was delivered using transfer protocols like ack/nak/timeout to insure the transfer did indeed happen.
KDS	Kitched Display System.
Kit	A collection of items that is priced & sold as a single item, but which is exploded into its constituent items for the purposes of tracking inventory. Typically used in retail
Layaway	A flavor of retail transaction line item whereby the customer is purchasing an item, but is not paying for it now. The store will put the item aside, and once the customer has completed payment they will take it with them.
Last activity date	The last date/time the SV has been used via any transaction type
Media Type	Paper, Card, SmartCard,. Virtual
Maximum recharge	The maximum amount that can be added to the value of a

Term	Definition
amount	SV account
OCB	Order Confirmation Board
On The Fly	As items are entered in the POS, they are immediately sent/made available to other systems, such as a KDS or OCB.
One Behind	As an item is entered in the POS, the previously entered item is sent/made available to other systems, such as a KDS or OCB.
On Total	All items as stored in the POS until the total key is pressed. At which point in time, they are sent/made available to other systems, such as a KDS or OCB.
Operator	The ID of the operator initiating the SV transaction
Over-ring	A register error in which a higher price (or quantity) than the actual price (quantity) of the goods is recorded in the register.
Party ID	Common element uniquely identifying a party (holder, collector, provider, operator, etc.)
POS	An acronym for Point of Service that includes traditional Point of Sale devices in addition to Kiosks, Websites and Telephone call centers and any other device or system that is used to record the exchange of Merchandise for Tender.
Point-of-Service (POS) Entry Mode	Indicates the method by which the PAN was entered, according to the first two digits of the ISO 8583:1987 POS Entry Mode
POSLOG	This refers to the Transaction Log containing all of the Retail Touch Point transactions. This log is typically transmitted to the HOST application where it is processed by an Enterprise level application. Archaic: An abbreviation for the POS Transaction Log, an application that stores transactions received from other applications for future reference, and makes them available to other applications in the system.
Provider	The ID of the party holding the liability for the SVI
RainCheck	A flavor of retail transaction line item whereby the customer wanted to purchase an item, but it is unavailable, and the store is issuing a special receipt guaranteeing the price of the item to the customer when it next becomes available.
Remaining Charges	The maximum number of times the account can be incremented. May be infinite
Return	A flavor of retail transaction line item whereby the customer is returning an item and they have bought it with them.
ReturnForDelivery	A flavor of retail transaction line item whereby the customer is returning an item and it will be delivered to some location at some future date & time.
ReturnForPickup	A flavor of retail transaction line item whereby the customer is returning an item and the store will pick it up from some location at some future date & time.
RTP	Retail touch point. This represents any location used as the source for selling, issuing or enquiring on Stored Value

Term	Definition
	Instruments.
RTP Transaction ID	The unique ID of the transaction on the RTP providing context of the SV transaction (Retailer, Store, Device, Sequence, Date/Time)
RTP Transaction Number	The unique transaction identifier generated and stored by the RTP for all transactions in which it participates. The RTP should maintain a cross-reference to the corresponding Stored Value Transaction number.
Sale	A flavor of retail transaction line item whereby a customer is purchasing an item and they are taking it with them.
SaleForDelivery	A flavor of retail transaction line item whereby a customer is purchasing an item and it will be delivered to some location at some future date & time.
SaleForPickup	A flavor of retail transaction line item whereby a customer is purchasing an item and they will pick it up from some location at some future date & time.
Start Date	The date the stored value account/instrument becomes valid for use.
Stored Value Instrument ID	A globally unique identifier associated with the instrument
Stored Value Instrument Media	The type of instrument media.- enumeration (paper, card, smart card, virtual)
Stored Value Issuer	The party that activated the instrument.
Stored Value Instrument Status	The current status of the account or instrument – (Stolen, active, inactive, overdrawn, expired, etc.)
SVA	Stored value application. This represents any application that stores, maintains and controls the information about Stored Value Accounts, and the associations between those accounts and Stored Value Instruments.
SVA Transaction ID	The ID assigned by the SVA to the SV transaction
Stored Value Transaction Number	The unique transaction identifier generated and stored by the Stored Value Application for all transactions in which it participates. The Stored Value Application should maintain a cross-reference to the corresponding RTP transaction number.
SV Transaction Type	The type of transaction performed on the SVA from the RTP – (activate, decrement, recharge, deactivate, reverse, inquiry)
Terminal Capabilities	Indicates the card data input, CVM, and security capabilities of the terminal
Terminal Type	Indicates the environment of the terminal, its communications capability, and its operational control
TranCryptogramType	TC – Transaction Certificate ARQC - Authorisation Request Cryptogram AAC - Application Authentication Cryptogram
Transaction Status Information	Indicates the functions performed in a transaction Offline data authentication was performed Cardholder verification was performed

Term	Definition
	Card risk management was performed Issuer authentication was performed Terminal risk management was performed Script processing was performed
Transaction Value	The amount and unit of measure of the value transacted with the SVA on the specified account
Terminal Verification Results	Status of the different functions as seen from the terminal
Under-ring	A register error in which a lower price (or quantity) than the actual price (quantity) of the goods is recorded in the register.
Unpredictable Number	Value to provide variability and uniqueness to the generation of a cryptogram
Value	The monetary or equivalent value managed by the stored value application.
Value Enumeration	Points, Miles, Kilometers, Minutes
Voucher	A sub-type of ControlTransaction that records the result of an interraction between the POS and StoredValue (voucher) management system. (cf. IXRetail StoredValue work-team & schema).
	A special Voucher for a specific product or service, that is issued or redeemed as part of a CustomerLoyalty scheme. Not to be confused with a GiftCertificate which has a monetary value