Electronic Invoicing Configuration Guide

SAP Point-of-Sale 2.3

Document Version: 1.9 – 2015-11-30

CUSTOMER

SAP POS 2.3 Electronic Invoicing Configuration Guide

Typographic Conventions

Type Style	Description
Example	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Textual cross-references to other documents.
Example	Emphasized words or expressions.
`EXAMPLE`	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE`	Keys on the keyboard, for example, `F2` or `ENTER`.

Document History

Version	Date	Change
1.0	2012-06.01	Initial version released for SAP POS 2.3 SP01 release.
1.1	2012-07-01	Added configuration for the feature on legal compliance of the government of Mexico requirement published version 2.2 of Diario Oficial de la Federación (CFD) on December 30th, 2011.
1.4	2012-11-30	Revised for SAP POS 2.3 SP04 release. Updated the screen graphics and procedures in section *5.8*, *Configuring SAP Point-of-Sale*.
1.7	2013-10-01	Added the enhanced *5.7*, *Configuring the EInvoice Receipt*.
1.8	2014-02-01	Updated to reflect changes to Mexico's legal requirements.
1.9	2015-11-30	Updated document template. Minor clarifications.

Contents

1 Intended Audience

This document is intended for:

- System administrators
- Solution consultants
- Technical consultants
- Support specialists
- Business process experts/owners

2 Glossary

Term	Definition
EInvoice (EI), also referred to as EInvoice 2.7	SAP-Developed Web application that provides a user interface for creating, printing, and canceling electronic invoices (EInvoices). SAP EInvoice 2.7 is a component of the SAP POS 2.3 suite.
POS terminal at store	Access point for electronic invoicing application where SAP POS 2.3 is running.
Third Party Vendor Components	Any applications or components that are not SAP applications or components. This could be a combination of the Third Party Web Service Façade Server and Third Party electronic invoicing application(s) or these could be combined as one.
Third Party Web Service Façade Server	A third-party web service application server that implements the server-side EInvoice Generation Interface and services SAP EInvoice 2.7. This application can also be used to implement the client-side EInvoice Management and Customer Management Interface. This application can interface with the third-party Electronic Invoicing application. This can be developed and provided by the retailer, third-party vendor, and/or SAP consulting. For more information, see section *7,Web Service Interface*.
Third Party electronic invoicing application(s)	This application(s) will consist of a third-party's component in servicing electronic invoicing processes. This application is the main engine in creating, and cancelling the electronic invoice that adheres to the Mexican government requirements. This can be developed and provided by the retailer, third-party vendor, and/or SAP consulting.

3 Getting Started

This document defines the configuration of the SAP EInvoice. It also covers the user functionalities of EInvoice, and technical reference of EInvoice.

3.1 Assumptions

This guide assumes that you have some level of familiarity with the terminology and functionality associated with SAP POS 2.3 and IBM Websphere 7.

This guide assumes that you have used the *SAP Returns Authorization 2.7 Installation Guide* to install both SAP Returns Authorization 2.7 and SAP EInvoice 2.7 available in SAP Service Marketplace at http://service.sap.com/instguides → *Industry Solutions* → *Industry Solution Guides* → *SAP for Retail* → *SAP Point-of-Sale* → *SAP POS 2.3.*

This guide is intended to provide you with the information you need to configure and interface with EInvoice, and navigate through and optimize the use of the EInvoice application in SAP POS. We recommend that you use this guide in conjunction with hands-on operation of the SAP POS 2.3 software.

This guide does not provide information about EInvoicing from a PC in a store where SAP POS 2.3 is not installed, or EInvoicing with customer self-service functions via the Internet.

4 Solution Overview

EInvoice 2.7 software was developed by SAP as an add-on to the SAP Point-of-Sale (SAP POS 2.3). It is a Web application that provides a user interface for creating, reprinting, and canceling electronic invoices for transactions generated in SAP POS 2.3. It is hosted at the head office with SAP Returns Authorization 2.7 and accessed from a POS terminal where SAP POS 2.3 is installed.

The following business processes are supported:

- Issuing and printing an EInvoice from an SAP POS terminal
- Reprinting an EInvoice from an SAP POS terminal
- Canceling an EInvoice from an SAP POS terminal

The store's employee performs these activities on SAP POS store terminals at stores where SAP POS 2.3 is installed.

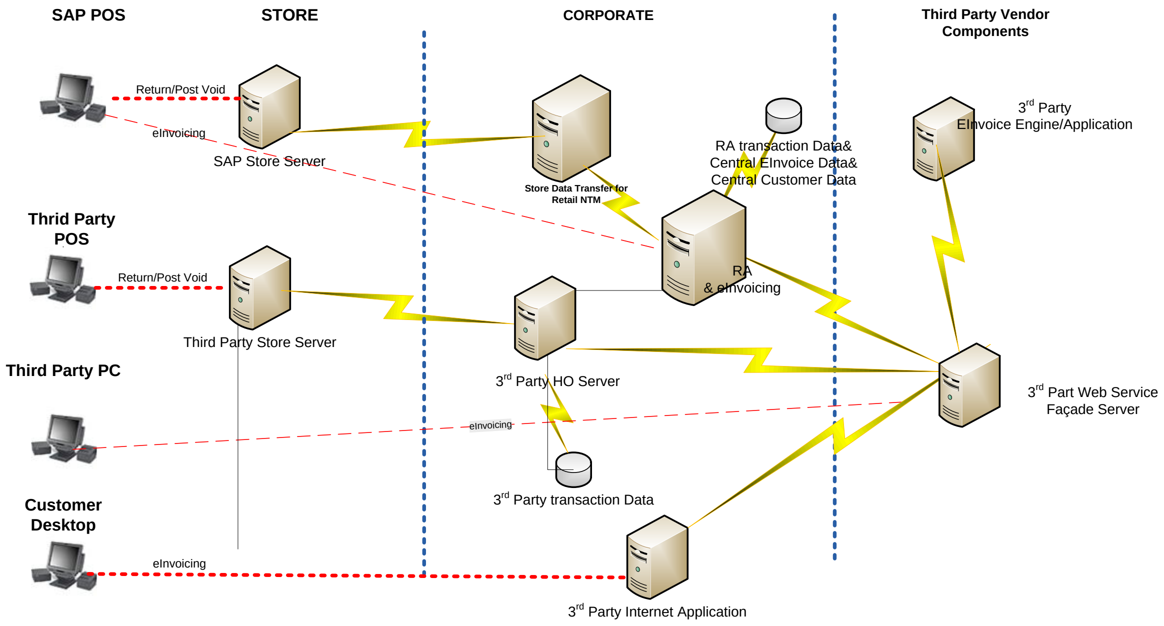
4.1 Software Component Matrix

The table below lists the business scenarios and the software components for the EInvoice application.

Scenario/Business Process	Software Component	Implementation and Configuration Process
Issuing and printing an electronic invoice	SAP POS 2.3 SAP EInvoice 2.7 SAP Returns Authorization 2.7 Third Party Web Service Façade Server Third Party Electronic Invoicing application	For more information, see section *5, Configuration*.
Reprinting an electronic invoice	SAP POS 2.3 SAP EInvoice 2.7 SAP Returns Authorization 2.7	For more information, see section *5, Configuration*.
Canceling an electronic invoice	SAP POS 2.3 SAP EInvoice 2.7 SAP Returns Authorization 2.7 Third Party Web Service Façade Server Third Party Electronic Invoicing application	For more information, see section *5, Configuration*.

4.2 Technical System Landscape

The figure below provides an overview of the system landscape. The EInvoice application is hosted on the same physical machine as the SAP Returns Authorization server.


SAP POS
STORE
CORPORATE
Third Party Vendor Components
Return/Post Void
eInvoicing
SAP Store Server
Store Data Transfer for Retail NTM
RA transaction Data& Central EInvoice Data& Central Customer Data
3rd Party EInvoice Engine/Application
RA & eInvoicing
Thrid Party POS
Return/Post Void
Third Party Store Server
3rd Party HO Server
3rd Part Web Service Façade Server
Third Party PC
eInvoicing
3rd Party transaction Data
Customer Desktop
eInvoicing
3rd Party Internet Application

4.3 Application Components

SAP Software	Purpose
SAP POS Store Data Transfer for Retail 1.10 (formerly known as Transnet and CEFT)	SAP application that supports the distribution of files and other resources throughout an internal network, allowing files to be managed through a central server with file distribution taking place from the server to a client or among clients. It handles the sending of transactions or messages from application-to-application between different hardware and software systems using multiple protocols.
SAP Point-of-Sale 2.3	SAP application for transaction and point-of-sale management for general merchandise retailers running client-server environments. It consists of store software (Store Manager, Xpress Server running in a store's back office); POS client software to process POS transactions; head-office software (Store Configurator) for central configuration.

SAP Software	Purpose
SAP Returns Authorization 2.7	SAP application for centralized returns authorization processing designed to work with SAP POS. The database contains sale and return transactions, pricing history, and business rules. The authorization component verifies return requests initiated at POS terminals and generates return approvals or referrals.

4.4 Third Party Vendor Components

Third Party Software	Purpose
Third Party Web Service Façade Server	A third-party web service application server that implements the server-side EInvoice Generation Interface and services SAP EInvoice 2.7. The vendor can also use this application to implement the client side EInvoice Management and Customer Management Interface. This application may interface with the third party electronic invoicing application. This can be developed and provided by the retailer, third-party vendor, and/or SAP consulting. For more information, see section *7*, *Web Service Interface*.
Third Party electronic invoicing application.	These applications consist of a third-party's component in servicing electronic invoicing processes. This application is the main engine in creating, and cancelling electronic invoicing that adheres to Mexican government requirements. This can be developed and provided by the retailer, third-party vendor, and/or SAP consulting.
Third Party POS System	The POS system comprised of a typical POS system such as POS client, store server, internet application and so on. This can interface with the third-party web service façade server which then interfaces with SAP EInvoice.

4.5 Electronic Invoicing Overview

Electronic invoicing refers to the process where a retail customer goes to a store with a transaction receipt and requests the creation of an electronic invoice (EInvoice) for certain, or all, items within a transaction to provide this information to the government to get a tax return. Store employees can perform the following activities on SAP POS store terminals at stores where SAP POS 2.3 is installed.

- Issuing an EInvoice from an SAP POS terminal
- Reprinting an EInvoice from an SAP POS terminal
- Canceling an EInvoice from an SAP POS terminal

5 Configuration

5.1 EInvoice Configuration Properties File

General application-level properties are contained in this file and are loaded once the electronic invoicing application is started.

`einvoiceConfig.properties` is stored under **`<SAP POS Returns Authorization installation folder>`**`\EInvoice\EInvoiceConfig` (for example, `C:\Program Files\SAP\ReturnsAuthorization\EInvoice \EInvoiceConfig`).

Any change to this file requires a restart of the EInvoice application, that is, the `EInvoiceEAR` and the `EInvoiceWSEAR` applications.

The following table describes the configuration properties for the EInvoice application.

Property Grouping	Property	Description
Currency Sign	`currency.sign`	Currency sign used in the EInvoice application. Default value: `$`
Date Format	`date_pattern`	Pattern value used for the actual user interface (UI) validation process. Default value: `ddMMyyyy`
	`date_pattern_lbl`	Pattern value displayed to the user in the UI. Default value: `DDMMYYYY`
Logging	`log.regular.path`	Relative path where the EInvoice logs are located (relative starting at `%WAS_HOME%\profiles\[PROFILE_NAME]`). Default value: `logs/server1/EInvoice.log`
	`log.regular.filesize`	Default file size for log files. Default value: `1000000`
	`log.regular.filecount`	Number of log files before logging starts overwriting the initial file. Default value: `5`
	`log.regular.level`	Default TLog level. Default value: `ALL`
	`exceptiondump.path`	Relative path where the failed EInvoices are dumped.

Property Grouping	Property	Description
Location of EInvoice Exception Dump Files		Default value: `logs/server1/failedEInvoices`
	`exceptiondump.file.extension`	Default file extension. Default value: `.xml`
EInvoice UI Configuration	`max_line_item_description_length`	Property for UI layout that controls the maximum display length of each item description in `eInvoiceCreateByItem` (item selection table in the UI). If an item description is longer than this value, the description is truncated, and a string continuation marker, such as `...`, is appended to the description. This marker is defined by the `string_continuation_marker` literal property. It does not affect the printed EInvoice; the entire full-length item description is printed on the EInvoice. Valid values: unsigned integer (`0`, `1`, `2`, `...`) Default value: `20`
	`max_customer_data_field_length`	Property for UI layout that controls the maximum displayed length of each customer information data element in the EInvoice confirmation page of the UI. If a data element is longer than this value, it is truncated and a string continuation marker, such as `...`, is appended. This marker is defined by the `string_continuation_marker` literal property. It does not affect the customer data entry fields or the printed EInvoice; the entire full-length customer information is printed on the EInvoice. Valid values: unsigned integer (`0`, `1`, `2`, `...`) Default value: `20`
	`customer_page_taxid_field_mandatory`	Property for the UI that controls whether a field is mandatory. Default value: `true`
	`customer_page_name1_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`

Property Grouping	Property	Description
	`customer_page_name2_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `false`
	`customer_page_streetname_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_apartmentnumber_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `false`
	`customer_page_streetnumber_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_neighborhood_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_city_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_county_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_state_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_country_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_postalcode_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `true`
	`customer_page_telephonenumber_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `false`
	`customer_page_email_field_mandatory`	Property for the UI that controls whether the field is mandatory. Default value: `false`
	`customer_page_reference_field_mandatory`	Property for the UI that controls whether the field is mandatory.

Property Grouping	Property	Description
		Default value: `false`
	`mandatory_field_marker`	Property for the UI indicating that data entry is mandatory for a field in the Enter Customer Information section of the Customer Information Entry page. Default value: `*`
	`customerInfo.postalCodeType`[1]	Property for the UI that controls the validation of the zip code entry in the Customer Information Entry page. Valid values: alpha (for letters only), alphanumeric (for letters and digits only), or numeric (for digits only). Default value: numeric
	`customerInfo.postalCodeLength`[1]	Property for the UI that controls the length of the zip code in the Customer Information Entry page. Valid values: unsigned integer (`0`, `1`, `2`, `...`) Default value: `5`
Codes Used Throughout the Applications	`eInvoice.srcSystemCode`	System that made a given Web Service call. Default value: `SAP-EI`
	`eInvoice.trxOriginCode`	System from which a given transaction originated. Default value: `SAP-RA`
	`eInvoice.customer.type`	Customer type value; should always be 01 (unless requirements change). Default value: `01`
Web Service URL	`eInvoiceGenerationWS.url`	URL to communicate with the EInvoice Generation Web Service Server Endpoint. Default value: *http://*`<IP_ADDRESS_HERE>`*/EInvoiceGenerationServer/EInvoiceGenerationService.asmx*
	`eInvoiceGenerationWS.documentType`	Document type value used when generating an EInvoice.

[1] If the postal code is not configured as a mandatory field (see `customer_page_postalcode_field_mandatory`), you can leave this field empty. However, if you specify a zip code, the validation rules for type and length are applied.

Property Grouping	Property	Description
		Default value: `FACTURA`
	`eInvoice.consumoItemSku`	Dummy consumo item SKU required by the EInvoice Generation Web service. Default value: `999999` ⚠ Caution Customer's EInvoice Generation Web Service implementation team must be advised if the default value is changed.
	`eInvoice.consumoUnitOfMeasure`	Default unit of measure for consumo items required by the EInvoice Generation Web Service. Default value: `SRV`
	`eInvoice.FiscalRegime`	Default text for fiscal regime that is used in the EInvoice receipts. Default value: `REGIMEN GENERAL DE LEY DE LAS PERSONAS MORALES`
	`eInvoice.PrintedCopy`	Default text for printed copy label that is used in the EInvoice receipts. Default value: `ESTE DOCUMENTO ES UNA REPRESENTACION IMPRESA DE UN CFDI`
	`tax.vat.id`	Tax ID that identifies the VAT tax. This property is used by EInvoice receipt. The tax ID defined in this property will not be listed in the taxable amount, and the tax amount. The item's VAT tax amount will be included in the item net price, item unit price, sub-total and grand total. The value of this property must be 4 characters and padded by zero. Example SAP POS define IEPS VAT tax ID as 2: `tax.vat.id= 0002`
	`masking.rule`	Masking rule for storing, printing and logging credit card numbers. The following example shows how the masking rule works:

Property Grouping	Property	Description

		Example
		For the credit card number `1234567890`
		o Masking All:
		`masking.rule=*`
		`Result = **********`
		o No Masking:
		`masking.rule==`
		`Result = 1234567890`
		o Suffix Only Masking:
		`masking.rule=#ssss`
		`Result = ######7890`
		o Prefix Only Masking:
		`masking.rule=pppp#`
		`Result = 1234######`
		o Prefix and Suffix Masking:
		`masking.rule=pppp#ssss`
		`Result = 1234##7890`
		o Default value: #ssss

Tax exempt configuration	Attributes used to create dummy tax exempt objects in the EInvoice application POS/RA. They do not provide any tax information when the line is tax-exempted. The application creates a dummy tax exempt object to solve issues on the `ITComplements` side. The tax rate and tax amount are both zero in this dummy object.	
	`tax.exempt.id`	Tax exempt type in the EInvoice application. Default value: `1`
	`tax.exempt.description`	Description of the tax exempt type in the EInvoice application; displayed on the UI and printed on the EInvoice. For example, exempt. Default value: `Exento`
	`html.character.set`	Character encoding (page character set) of the Latin alphabet used by components of Microsoft Windows. SAP POS uses windows-1252 as the default character set. Default value: `windows-1252`

Property Grouping	Property	Description
	`http.request.monitor.enable`	Indicates if debugging is enabled to monitor the concurrent request bug in `JSF1.1`. Default value: flag is set to disabled (=`FALSE`); it is enabled (=`TRUE`) only for debugging.
Returns Authorization (RA)EJB JDNI Lookup Attributes	`ra.ejb.ip`	IP address of the EInvoice Server. Default value: `localhost`
	`ra.ejb.port`	Port number of the EInvoice Server for the remote EJB connection. Default value: `2809`
	`ra.ejb.jndi.name`	JNDI name of the remote EJB. Default value: `AuthEngine`
	`ra.integration.enable`	Indicates if transaction data is fetched from the SAP Returns Authorization application or dummy data is used. Default value: `false`
	`ra.version`	Version of SAP Returns Authorization application. Default value: `2.6.1`
	`ra.performance.test.flag`	Indicates if performance testing is executed on the rig where the EInvoice application is deployed. Default value: flag is set to disabled (=`FALSE`); it is enabled (=`TRUE`) only for performance testing. Default value: `false`
	`ra.sample.response`	Dummy transaction data returned by the RA application. The backslash character is used at the end of each line to include the lines in the property value. Note This property is only used for testing purposes. It requires the property ra.integration.enable to be set to (=`FALSE`).
	`tax.generalnational.ids`	List of general national tax IDs. Tax IDs are separated by a ',' if there is more than one.

Property Grouping	Property	Description
		Example XAXX010101000,XAXX01010100 1 Default value: XAXX010101000
	tax.generalforeign.ids	List of general foreign tax IDs. Tax IDs are separated by a ',' if there is more than one. Example XEXX010101000,XEXX01010100 1 Default value: XEXX010101000
	navigation.regular_invoice.skipitemselection	Indicates if search results and the item selection page should be skipped in a regular invoice printing scenario. Default value: true
	navigation.quick_invoice.skipitemselection	Indicates if search result and the item selection page should be skipped in a quick invoice printing scenario. Default value: true
	navigation.search_by_folio	Indicates if the Folio fields or transaction fields are displayed by default for a reprint and cancel scenario. Default value: false
	test.testmode	Indicates if the application is running in test mode. To be used only for debugging. When this parameter is set to (=TRUE), test data is used and no EJB methods are executed. Default value: false
	eInvoice.trx.search.attempts	In the quick EInvoice scenario, the number of attempts to search a transaction from the SAP Returns Authorization application. This is required because of the delay in the transaction transfer from SAP POS to SAP Returns Authorization. This value should be >= 1 and <= 5. Default value: 3
	eInvoice.trx.search.interval	Interval value, in seconds, between transaction search attempts. Must be >= 1 and <=5.

Property Grouping	Property	Description
		Default value: 1

In the configuration file you can change the default attributes of the UI fields, determine what UI fields are mandatory, and specify the character used to designate a mandatory field.

The **<…>**_field_mandatory properties listed in the table above control which customer fields are mandatory in the Customer Information Entry page. All mandatory fields in this page are designated by a mandatory field marker, such as *, after the field name.

For more information about the Customer Information Entry page, see section *8.8*, *Generating and Printing an EInvoice from a POS Terminal*.

5.2 EInvoice Format Configuration Properties

The component that manages the formatting of the generated EInvoice accesses this properties file. Sections of the EInvoice are configurable via a properties file shown below.

einvoiceFormatConfig.properties is stored under **<SAP POS Returns Authorization install folder>**\EInvoice\EInvoiceConfig (for example C:\Program Files\SAP\ReturnsAuthorization\EInvoice \EInvoiceConfig)

Any change to this file requires a restart of the EInvoice application, that is, the EInvoiceEAR and the EInvoiceWSEAR applications.

The following table describes the formatting parameters for the EInvoice application.

Parameter	Description
format_order	Indicates the order of the various sections of the EInvoice receipt. The value of format_order can be one or more of the Well-Known Form names. See section *10.1*, *List of Well-Known Forms*. Possible Value: HEADER, COMPANY_INFO, STORE_INFO, FISCAL_REGIME, FOLIO_INFO, ISSUE_INFO, CUSTOMER_INFO, TRX_DATA, ITEM_HEADER, REGULAR_ITEMS_DETAIL, FB_ITEMS_DETAIL, ITEM_DATA, ITEM_FOOTER, TAXABLE_SUBTOTAL_DATA, SUBTOTAL_GENERAL, TAX_TOTALS_DATA, TOTAL, TENDER_TYPE, FIXED_TEXT, APPROVAL_DATA, OIM, DIGITAL_SEAL
einvoice_begin_data	The generated EInvoice receipt is embedded in the Generate EInvoice or Reprint EInvoice HTML Web page as an HTML comment. The einvoice_begin_data configuration parameter marks the start of the generated EInvoice receipt within the HTML comment. Default Value: EINVOICE_BEGIN
einvoice_end_data	The generated EInvoice receipt is embedded in the Generate EInvoice or Reprint EInvoice HTML Web page as a HTML comment. The einvoice_end_data configuration parameter

Parameter	Description
	marks the end of the generated EInvoice receipt within the HTML comment. Default Value: `EINVOICE_END`
`total_line_char`	Total number of characters in a single line of the printed EInvoice. Default Value: `42`
`section_char`	Character used in the separation line in the printed EInvoice. Default Value: – Note This property is deprecated.
`date_pattern`	Format of the business date in the printed EInvoice. Default Value: `dd/MM/yyyy`
`time_pattern`	Format of the transaction time in the printed EInvoice. Default Value: `HH:mm:ss`
`FB_dept`	Comma separated list of unique identifiers of the food and beverage (F&B) departments. Each identifier is formed by a combination of the company ID (mandatory), store ID (optional), and department ID (mandatory). Syntax `FB_dept=`**`<companyID>`**`:`**`<storeID>`**`:`**`<deptID>`** ,… Where: **`<companyID>`** is the company ID. **`<storeID>`** is the store ID. **`<deptID>`** is the department ID. Example `FB_dept=1:1:7001,1::7002`
`doubleHighWideChar`	Double high wide character printer code. This printer code is used in the HEADER section. Default Value: In Hex: `0x1b0x7c0x320x43` In ASCI: `|2C`

5.3 EInvoice Receipt Document Form File

The `eInvoiceReceiptDocument.form` file contains the EInvoice receipt document configuration. The `eInvoiceReceiptDocument.form` file is a UTF-8 encoded file type.

Note

The UTF-8 encoded file allows you to enter Spanish (Mexico) text.

The configuration is grouped by a series of forms. Forms (also referred to as “document forms”) are similar to procedures that contain simple expressions and statements that allow you to format text and values and place them in the content of the receipt.

Two types of forms:

Form Type	Description
Well-Known Form	A Well-Known Form is a type of form that is handled and used by the EInvoice Receipt with special processing. Well-Known Forms identify an EInvoice receipt section. Well-Known Forms will only be activated and processed by EInvoice Receipt if listed in the `format_order` property of the `eInvoiceFormatConfig.properties` file.
User Defined Form	User Defined Forms are created by the user in order to define some part of the receipt content. They are typically created either to allow a part of the receipt content to be reused in multiple locations on the receipt, or to allow the content to be included on the receipt only under some conditions.

The form consists of the following form components:

Form Component	Description
Form Name	A unique identifier for the form. The Form Name is prefixed with a period (`.`) character. For example: `.HEADER`. Note Form Names are case-sensitive.
Form Contents	Defines the contents of the form. The form contents can consist of the following: • EInvoice Variables: Data input from the POS system and output of information generated by the EInvoice • User Text: Any alphanumeric characters • Functions: String and numeric manipulation functions • Form References: Allows insertion of a user defined form within another form. (via `.insert` keyword)
Form Conditional Statement	An expression that produces a `true` or `false` result. If the result is `true`, the form is processed and its contents are included in the receipt. If the expression is `false`, the next form in the chain (if any) is processed.

Form Component	Description
Comment	Any lines starting with the hash (#) character will be treated as a comment.

Functions

Note

Functions are case-insensitive.

Functions	Description

`Left`	Returns the leftmost part of the value.
	Syntax
	`~left(value, length)`
	Where:
	`value` is the input string to parse.
	`length` is the number of characters to return from the left end of the input string.
	Note
	○ If the specified length is less than or equal to zero, the result will be an empty string.
	○ If the specified length is greater than the length of the specified input string, the result will be the specified input string.
	Example
	`value=”Configurable EInvoice Receipt”`
	`~left(%value%, 12)`
	Result: `Configurable`

`Right`	Returns the rightmost part of the value. Syntax `~right(value, length)` Where: `value` is the input string to parse. `length` is the number of characters to return from the right end of the input string. Note ○ If the specified length is less than or equal to zero, the result will be an empty string.

Functions	Description

	○ If the specified length is greater than the length of the specified input string, the result will be the specified input string.
	Example
	`value=”Configurable EInvoice Receipt”`
	`~right(%value%, 7)`
	Result: `Receipt`

`Mid`	Returns the middle of the value.
	Syntax
	`~mid(value, start, length)`
	Where:
	`value` is the input string to parse.
	`start` is the starting index. The first index is one.
	`length` is the number of characters to return from the input string, starting at the specified start index.
	Note
	○ If the specified start index is less than or equal to zero, the result will be the specified input string.
	○ If the specified start index is greater than the length of the specified input string, the result will be an empty string.
	○ If the specified length is less than zero, the result will be an empty string.
	○ If the specified length is zero, the result will be the substring of the specified input string starting from the specified start index.
	○ If the sum of the specified length and specified start index is greater than the length of the specified input string, the result will be the substring of the specified input string starting from the specified start parameter.
	Example
	`value=”Configurable EInvoice Receipt”`
	`~mid(%value%, 14, 8)`
	Result: `EInvoice`

`Length`	Returns the number of characters. Syntax `~length(value)` Where:

Functions	Description

	`value` is the input string.
	Example
	`value=”Configurable EInvoice Receipt”`
	`~length(%value%)`
	Result: `29`

`Trim`	Returns a copy of a string with leading and trailing white space omitted. A white space is any character less than \u0021 (ASCII table).
	Syntax
	`~trim(value)`
	Where:
	`value` is the input string.
	Example
	`value=” Configurable EInvoice Receipt ”`
	`~trim(%value%)`
	Result: `Configurable EInvoice Receipt`

`Wrapcentrealign`	Returns the center aligned specified string value. The alignment is based on the configured total_line_char property of the `eInvoiceFormatConfig.properties`. If the specified string value parameter is longer than the `total_line_char` property, the string is divided into two or more lines. Each line will be aligned to the center.
	Syntax
	`~wrapcentrealign(value)`
	Where:
	`value` is the input string.
	Example
	`value=”Configurable EInvoice Receipt”, and total_line_char is 42 characters.`
	`~wrapcentrealign(%value%)`
	Result: `      Configurable EInvoice Receipt`
	Example
	`value=”Very Long Lines Configurable EInvoice Receipt”, and total_line_char is 42 characters.`
	`~wrapcentrealign(%value%)`
	Result: `Very Long Lines Configurable EInvoice`

Functions	Description
	`Receipt`

`Wrapleftalign`	Returns the left aligned specified string value parameter. The alignment is based on the configured total_line_char property of the `eInvoiceFormatConfig.properties`. If the specified string value parameter is longer than the `total_line_char property`, the string is divided into two or more lines. Each line will be aligned to the left.
	Syntax
	`~wrapleftalign(value)`
	Where:
	`value` is the input string.
	Example
	`value="Configurable EInvoice Receipt", and total_line_char is 42 characters.`
	`~wrapleftalign(%value%)`
	Result:`Configurable EInvoice Receipt`
	Example
	`value="Very Long Lines Configurable EInvoice Receipt", and total_line_char is 42 characters.`
	`~wrapleftalign(%value%)`
	Result:`Very Long Lines Configurable EInvoice Receipt`

Special Character

Symbol	Description

Period Character (`.`)	The period character is used in the form name. Any line that starts with a period is expected to be followed by a form name.
	Use the following syntax to print a period character:
	Syntax
	`^.:x^`
	Where:
	`.` is the period character.
	`x` is the number of times to print.
	Example
	`^.:1^`
	Result: `.` (1 period)

Symbol	Description
	Example `^.:10^` Result: `..........` (10 periods)

Caret Character (`^`)	The caret character is used to print a character a specified number of times.
	Use the following syntax to print a caret character:
	Syntax
	`^^:x^`
	Where:
	^ is the caret character.
	x is the number of times to print.
	Example
	`^^:1^`
	Result: `^` (1 caret)
	Example
	`^^:10^`
	Result: `^^^^^^^^^^` (10 caret)

Hash Character (`#`)	The hash (number sign) character is used to identity a comment.
	Use the following syntax to print a hash character.
	Syntax
	`^#:x^`
	Where:
	`#` is the hash character.
	`x` is the number of times to print.
	Example
	`^#:1^`
	Result: `#` (1 hash)
	Example
	`^#:10^`
	Result: `##########` (10 hash)

Percent Character (`%`)	The percent character is used to identity a variable by enclosing the variable name with percent characters, for example `%trxStoreId%`. Use the following syntax to print a percent character:

Symbol	Description
	Example %% Result: %
Dollar Character ($)	Use the following syntax to print a dollar character. Example $$ Result: $
Tilde Character (~)	Use the following syntax to print a tilde character: Example ~~ Result: ~

Special Document Formatting Properties

Form Type	Description

Aligning Text	The pipe character, or vertical bar |, specifies text alignment.
	A single bar, or |, specifies left and center alignment (any text preceding the single pipe is left-aligned, and text following the pipe is centered).
	A double pipe, or ||, specifies right alignment.
	When no alignment is specified, the default is left alignment.
	Example
	`|Thank you for shopping at`
	Result: `Thank you for shopping at`
	Explanation: The line is centered.
	Example
	`||SAP`
	Result: `SAP`
	Explanation: The line is right aligned.

Aligning and Padding Variable	The semicolon character after the variable name provides the ability to align and pad the variable. The variable can be padded with spaces or with zeros. Syntax `%<variable_name>:<alignment><0><width>%` Where: `<variable_name>` refers to the variable name.

Form Type	Description

	`<alignment>` refers to the alignment type. Possible values are:
	○ `r` indicates right alignment
	○ `l` indicates left alignment. By default the alignment is left, hence, this is optional.
	○ `c` indicates center alignment.
	`<0>` indicates the variable will be padded with zero. This zero-pad indicator is optional and only works on right alignment. Hence, if the zero-pad is specified in a right alignment type, the variable will be padded with zeros, otherwise the variable will be padded with spaces.
	`<width>` specifies the number of characters the variable can cover.
	Caution
	If the specified width is less than the length of the variable's value, the value will be trimmed.
	Example
	With `a_variable = "Variable Alignment"`
	Example 1:
	`%a_variable:r20%`
	Result: `Variable Alignment`
	Explanation: The value is padded with two spaces, since the length of `a_variable` is 18. This will right align the variable.
	Example 2:
	`%a_variable:r020%`
	Result:`00Variable Alignment`
	Explanation: The value is padded with two zeros on the left side of the value, since the length of `a_variable` is 18.
	Example 3:
	`%a_variable:c20%`
	Result: `Variable Alignment`
	Explanation: The value is padded with one space on both the left and right side.
	Example 4:
	`%a_variable:c020%`
	Result: `Variable Alignment`
	Explanation: The value is padded with one space on both the left and right side. The zero-pad is ignored in this case since zero-pad only works on right alignment.

Form Type	Description

	Example 5:
	%a_variable:l020%
	Result:`Variable Alignment`
	Explanation: The value is padded with two spaces on the right side of the value, since the length of `a_variable` is 18. The zero-pad is ignored in this case since zero-pad only works on right alignment.
	Example 6:
	`%a_variable:20%`
	Result:`Variable Alignment`
	Explanation: The value is padded with two spaces on the right side of the value, since the length of `a_variable` is 18.
	Example 7:
	`%a_variable:8%`
	Result:`Variable`
	Explanation: The final value is trimmed, since the specified width (8) is less than the length (18).

Adding blank lines	Insert as many blank lines as required anywhere in the document form using the `.blank` statement.
	To specify more than one consecutive blank line, add a numeric value after the blank format.
	Example
	`.blank`
	Result: (1 blank lines)
	Example
	`.blank8`
	Result: (8 blank lines)

Specifying conditions for printing functions	This is known as a form conditional statement. The form conditional statement is an expression that produces a `true` or `false` result. If the result is `true`, the form is processed and its contents are included in the receipt. If the expression is `false`, the next form in the chain (if any) is processed. A condition can be specified as follows: Syntax `.<formname>,<condition(s)>,onfail=<linkeddocument>` Where:

<table>
<tr><th>Form Type</th><th>Description</th></tr>
<tr><td></td><td>**<formname>** refers to the name of the specified document form.
<linkeddocument> identifies an alternate user defined form to be processed when the condition(s) are not met.
<condition(s)> contains the list of conditions. This statement is optional. If used, you can define a document to print certain elements only when specified conditions are met, or specify another form or action when a condition fails. If more than one condition is needed, each condition must be separated by comma.
The following comparison signs can be used to express conditions: <, <=, =, >=, >, <>.
There are two types of conditions:
o Implicit condition: The condition is met when all variables used in the form are found. As long as a value exists, the form is rendered.
o Explicit condition: The value of the variable must exactly match the condition, for example count=1.
Condition results can be expressed in any of the following formats:
o Text value (%variable%=value)
o Numeric value (%variable%=value)
o Length tests (%variable%>Ln), where n is the length.
Example
Implicit Condition Example
.myformname,onfail=linkeddocument
%variable1%||%variable2%
Text for myformname
.linkeddocument
Text for linkeddocument
%variable3%||%variable4%
In this implicit condition example, if variable1 and variable2 exist, the myformname form will be rendered. If variable3 and variable4 exist, the linkeddocument form will be rendered. If neither condition is met, nothing will be rendered.
Example
Explicit Condition Example
.myformname,~length(%variable1%)>10, %variable2=2,onfail=linkeddocument
%variable1%||%variable2%
Text for myformname
.linkeddocument</td></tr>
</table>

Form Type	Description

	`Text for linkeddocument`
	`%variable3%||%variable4%`
	In this explicit condition example, if both conditions `(~length(%variable1%)>10`, and `%variable2=2`) are met the `myformname` form is rendered, otherwise `linkeddocument` form is rendered.

Inserting a user defined form	Ability to insert user defined form within any form is provided.
	Syntax
	`.insert <user define form name>`
	Example
	`.TENDER_TYPE`
	`Payment: %tenderId%`
	`Account: %tenderAccountNumber%`
	`.insert TENDER_AMOUNT`
	`.blank`
	`.TENDER_AMOUNT`
	`Amount: $tenderAmount%`

Example

Sample EInvoice Receipt Document:

```
#*****************************************************************************
#* THIS OPEN SOURCE SOFTWARE IS DISTRIBUTED IN THE HOPE THAT IT WILL BE USEFUL, BUT
#* IS PROVIDED "AS IS" WITHOUT ANY EXPRESSED OR IMPLIED WARRANTY; INCLUDING BUT NOT
#* LIMITED TO THE IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
#* PURPOSE. IN NO EVENT SHALL SAP, THE COPYRIGHT HOLDERS, OR THE CONTRIBUTORS BE
#* LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
#* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES
#* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
#* THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
#* NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
#* IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
#*****************************************************************************


# EInvoice Receipt Document Configuration
# This file contains the EInvoice receipt document layout configuration.  This file is
UTF-8 encoded file type.
```

```
# The file is grouped by form.  A form is marked by period (.) followed by the form name.  With the exception of .insert and .blank keywords
# The order of form processing is configured in format_order property found in eInvoiceFormatConfig.properties file.

# HEADER form is a Well-Known Form
# Each line in this HEADER form is printed with double high wide, and center aligned.
# Prints the Company name
.HEADER
|%storeCompanyName1% %storeCompanyName2%|

# COMPANY_INFO form is a Well-Known Form
# Prints the company address and contact info
.COMPANY_INFO
~WRAPCENTREALIGN(%storeCompanyAddressStreetName% %storeCompanyAddressStreetNumber%%COMMA%%storeCompanyAddressApartmentNumber%%COMMA%%storeCompanyAddressNeighborhood%%COMMA%%storeCompanyAddressCounty%)
~WRAPCENTREALIGN(%storeCompanyAddressCity%%COMMA%%storeCompanyAddressState%%COMMA%%storeCompanyAddressCountry%%COMMA%%zipcode_lbl%%storeCompanyAddressPostalCode%)
|%telephone_lbl%%storeCompanyAddressTelephoneNumber%|
|%taxID_lbl% %storeCompanyTaxId%|
.blank

# STORE_INFO form is a Well-Known Form
# Prints the store address and contact info
.STORE_INFO
~WRAPCENTREALIGN(%storeName1% %storeName2%)
~WRAPCENTREALIGN(%storeAddressStreetName% %storeAddressStreetNumber%%COMMA%%storeAddressApartmentNumber%%COMMA%%storeAddressNeighborhood%%COMMA%%storeAddressCounty%)
~WRAPCENTREALIGN(%storeAddressCity%%COMMA%%storeAddressState%%COMMA%%storeAddressCountry%%COMMA%%zipcode_lbl%%storeAddressPostalCode%)
|%taxID_lbl% %storeTaxId%|
^-:42^

# FISCAL_REGIME form is a Well-Known Form
# Prints fiscal regime text.
.FISCAL_REGIME
~WRAPLEFTALIGN(%fiscal_regime_lbl% %storeCompanyFiscalRegime%)
^-:42^

# FOLIO_INFO form is a Well-Known Form
# Prints folio info
.FOLIO_INFO
%folio_no_lbl%%folioPrefix% %folioNumber%

# ISSUE_INFO form is a Well-Known Form
# Prints issue info
.ISSUE_INFO
%issue_place_date_lbl%
%storeAddressCity%%COMMA%%storeAddressState% %invoiceDate%
^-:42^

# CUSTOMER_INFO form is a Well-Known Form
# Prints customer's address info
.CUSTOMER_INFO
~WRAPLEFTALIGN(%customer_lbl% %customerName1% %customerName2%)
%taxID_lbl% %customerTaxId%
~WRAPLEFTALIGN(%customerAddressStreetName% %customerAddressStreetNumber%%COMMA%%customerAddressApartmentNumber%%COMMA%%customerAddressNeighborhood%%COMMA%%customerAddressCounty%)
~WRAPLEFTALIGN(%customerAddressCity%%COMMA%%customerAddressState%%COMMA%%customerAddressCountry%%COMMA%%zipcode_lbl%%customerAddressPostalCode%)
^-:42^

# TRX_DATA form is a Well-Known Form
# Print the sale trx info
.TRX_DATA
%date_lbl%%trxBusinessDate%||%time_lbl%%trxBusinessTime%
%trx_lbl%%trxNumber%||%store_fmt_lbl%%trxStoreId%
%cashier_lbl%%trxCashierId%||%terminal_lbl%%trxRegisterId%
^-:42^

# ITEM_HEADER form is a Well-Known Form
# Prints item header labels
.ITEM_HEADER
%quantity_lbl%     %description_lbl%
     %unit_price_lbl%||%amount_lbl%

# REGULAR_ITEMS_DETAIL form is a Well-Known Form
# This form is used and processed for each invoiced regular item in the transaction.
# Prints item info for the current item. If the customer tax ID is general national tax ID the gross price is used; otherwise net price is used.
```

```
.REGULAR_ITEMS_DETAIL,%GENERAL_NATIONAL_TAX_IDS%=%customerTaxId%,onfail=notNationalGenericTaxForRegularItem
~WRAPLEFTALIGN(%qty:r3% %measuringUnit:r5% %sku% %description%)
     %grossUnitPrice%||%grossPrice%

.notNationalGenericTaxForRegularItem
~WRAPLEFTALIGN(%qty:r3% %measuringUnit:r5% %sku% %description%)
     %unitPrice%||%netPrice%

# ITEM_FOOTER form is a Well-Known Form
# Prints if exist food and beverage item.
.ITEM_FOOTER,%GENERAL_NATIONAL_TAX_IDS%=%customerTaxId%,%fbItemExist%=true,onfail=notNationalGenericTaxForFBItemFooter
%EINVOICE_CONSUMO_UNIT_OF_MEASURE:r8% %food_item_text%
||%fbItemGrossPriceSum%
^-:42^

.notNationalGenericTaxForFBItemFooter,%fbItemExist%=true,onfail=noFBItemFooter
%EINVOICE_CONSUMO_UNIT_OF_MEASURE:r8% %food_item_text%
||%fbItemNetPriceSum%
^-:42^

.noFBItemFooter
^-:42^



# TAXABLE_SUBTOTAL_DATA form is a Well-Known Form
# This form is used and processed for each taxable subtotal in the transaction.
# Prints the each taxable subtotal
.TAXABLE_SUBTOTAL_DATA
%subtotal_lbl% %taxableSubTotalDesc%||%taxableSubTotalAmount%

# SUBTOTAL_GENERAL form is a Well-Known Form
# Prints EInvoice grand total amount if the customer tax ID is general national ID; otherwise, prints the EInvoice net amount
.SUBTOTAL_GENERAL,%GENERAL_NATIONAL_TAX_IDS%=%customerTaxId%,onfail=notNationalGenericTaxForSubtotalGeneral
^-:42^
%subtotal_general_lbl%||%eInvoiceGrandTotalAmount%
```

```
.notNationalGenericTaxForSubtotalGeneral
^-:42^
%subtotal_general_lbl%||%eInvoiceNetAmount%

# TAX_TOTALS_DATA form is a Well-Known Form
# This form is used and processed for each tax total data in the transaction.
# Prints the each tax total amount if customer tax ID is not general national tax ID;
otherwise, print zero tax amount
.TAX_TOTALS_DATA,%GENERAL_NATIONAL_TAX_IDS%=%customerTaxId%,onfail=notNationalGenericT
axForTaxTotalData
%taxTypeSubTotalTaxInfoTaxDesc%||%ZERO_AMOUNT%

.notNationalGenericTaxForTaxTotalData
%taxTypeSubTotalTaxInfoTaxDesc%||%taxTypeSubTotalTaxInfoTaxAmount%

# TOTAL form is a Well-Known Form
# Prints the grand total amount, and its string representation
.TOTAL
%total_lbl%||%eInvoiceGrandTotalAmount%
~WRAPLEFTALIGN((%amountOnText%))
^-:42^

# TENDER_TYPE form is a Well-Known Form
# This form is used and processed for tender in the transaction.
# Prints tender description and account number (if exist)
.TENDER_TYPE
%tender_desc_lbl% %tenderDescription%
%tender_acct_num_lbl% %tenderAccountNumber%

# FIXED_TEXT form is a Well-Known Form
# Print some fix text.
.FIXED_TEXT
^-:42^
~WRAPLEFTALIGN(%fixed_text_1%)
~WRAPLEFTALIGN(%fixed_text_2%)
^-:42^

# APPROVAL_DATA form is a Well-Known Form
# Prints EInvoice approval information
.APPROVAL_DATA
%approval_no_lbl%%approvalNumber%
```

```
%approval_year_lbl%%approvalYear%
%certicate_no_lbl%%certificateNumber%
.blank

# OIM form is a Well-Known Form.
# Prints original input message (OIM) info
.OIM
~WRAPLEFTALIGN(%original_message_lbl%%originalInputMessage%)
.blank

# DIGITAL_SEAL form is a Well-Known Form
# Prints digital deal
.DIGITAL_SEAL
~WRAPLEFTALIGN(%digital_seal_lbl%%digitalSeal%)
```

5.4 EInvoice Checking Rules

Checking rules are rules against which a transaction is checked before an EInvoice is generated and printed. If the transaction fails any one of the rules, the transaction cannot be invoiced. Checking rules exist also for the line items within a transaction. If a line item fails any one of the rules, that line item cannot be invoiced.

You can configure a subset of the checking rules, which are known as configurable parameter rules. In this section we focus on the rules that you can configure.

Note

- All configurable parameter rules are optional. That is, if you do not want a checking rule (parameter), you do not include the rule in the configuration file.
- There are no defaults for configurable parameter rules.
- You configure the configurable parameter rules only in a negative way. That is, if the rule is satisfied, the EInvoice action is rejected on either the transaction level or item level.

5.4.1 Configurable Parameter Rules

Configurable parameter rules are defined in an XML file (see the example below).

This XML file is stored at **`<SAP POS Returns Authorization installation folder>`**`\EInvoice\EInvoiceConfig\ eInvoiceCheckingRuleConfig.xml` (for example, `C:\Program Files\SAP\ReturnsAuthorization\EInvoice \EInvoiceConfig\ eInvoiceCheckingRuleConfig.xml`).

Any change to this file requires a restart of the EInvoice application, that is, the `EInvoiceEAR` and the `EInvoiceWSEAR` applications.

The following table describes the rules that you can configure.

Parameter	Description
Expiration Day	The maximum period of days from the day of transaction within which an EInvoice can be generated. Syntax `<ExpirationDay>30</ExpirationDay>` Example Expiration Day =30 If a transaction takes place on August 2, 2010, the customer has 30 days after that transaction date to request an EInvoice. If the customer comes to the store on September 10, 2010, and requests an EInvoice, the transaction cannot be invoiced as 30 days have expired. If the customer comes to the store on August 10, 2010, and requests an EInvoice, the transaction can be invoiced as 30 days have not expired. Example Expiration Day =0 Invoicing can take place only on the same day of the transaction, in this example on August 2, 2010. In other words, the customer cannot come to the store on August 3, 2010, and request an invoice.

Item Department	If a line item has an item department that appears in this configuration file, then it cannot be invoiced.
	Syntax
	`<ItemDepartmentList>`
	`<ItemDepartment>InvalidDep001</ItemDepartment>`
	`<ItemDepartment>InvalidDep002</ItemDepartment>`
	`</ItemDepartmentList>`

Item SKU	If a line item has an item SKU that appears in this configuration file, then it cannot be invoiced.
	Syntax
	`<ItemSKUList>`
	`<ItemSKU>InvalidSku001</ItemSKU>`
	`<ItemSKU>InvalidSku002</ItemSKU>`
	`</ItemSKUList>`

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>Tender</td><td>

A tender is identified by a combination of the `StoreID` and `TenderID`. If a tender in a transaction has a combination of the `StoreID` and `TenderID` that appears in this configuration file, then it cannot be invoiced.

Syntax

```
<TenderList>
   <Tender>
     <StoreID>111</StoreID>
<TenderID>InvalidTenderPennies</TenderID>
   </Tender>
   <Tender>
     <StoreID>222</StoreID>
<TenderID>InvalidTenderLoonies</TenderID>
   </Tender>
</TenderList>
```

To specify that a particular `TenderID` (for example, `InvalidTenderEuros`) is invalid for invoicing at all stores, specify an asterisk (`*`) for the value of `StoreID`.

Syntax

```
<TenderList>
   <Tender>
     <StoreID>*</StoreID>
<TenderID>InvalidTenderEuros</TenderID>
   </Tender>
   <Tender>
     <StoreID>222</StoreID>
<TenderID>InvalidTenderLoonies</TenderID>
   </Tender>
   </TenderList>
```

</td></tr>
</table>

5.4.2 Configurable Parameter Rules File

Example

```
<?xml version="1.0" encoding="UTF-8" ?>
- <EInvoiceCreationRules xmlns:n1="http://www.sap.com/POS/EInvoice/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
    <ExpirationDay>0</ExpirationDay>
  - <ItemDepartmentList>
    <ItemDepartment>InvalidDep001</ItemDepartment>
    <ItemDepartment>InvalidDep002</ItemDepartment>
    </ItemDepartmentList>
  - <ItemSKUList>
    <ItemSKU>InvalidSku001</ItemSKU>
    <ItemSKU>InvalidSku002</ItemSKU>
    </ItemSKUList>
  - <TenderList>
  - <Tender>
    <StoreID>111</StoreID>
    <TenderID>InvalidTenderPennies</TenderID>
    </Tender>
  - <Tender>
    <StoreID>*</StoreID>
    <TenderID>InvalidTenderEuros</TenderID>
    </Tender>
    </TenderList>
    </EInvoiceCreationRules>
```

5.5 Master Data

The master data file provides the company and store data that is used to print the EInvoice. An excerpt of the file is shown below.

The SAP POS terminal provides the valid company ID and store ID to the EInvoice application, which uses these IDs to locate the complete company and store information in this file. When the name, tax ID, or address of a given store or the company head office changes, you must update the master data file.

The masterData.xml file is stored under **`<SAP POS Returns Authorization installation folder>`**`\EInvoice\EInvoiceConfig` (for example, `C:\Program Files\SAP\ReturnsAuthorization\EInvoice \EInvoiceConfig`).

Any change to this file requires a restart of the EInvoice application, that is, the `EInvoiceEAR` and the `EInvoiceWSEAR` applications.

5.5.1 Master Data File

Example

```
<Company id="33">
```

```
<Name>ADMINISTRACION INTEGRAL DE ALIMENTOS S.A. DE C.V.</Name>
<TaxId>AIA941123Q99</TaxId>
<Street>CALVARIO</Street>
<StreetNumber>106</StreetNumber>
<ApartmentNumber />
<Reference />
<Neighborhood>COL. TLALPAN CENTRO</Neighborhood>
<County>DELEG. TLALPAN</County>
<PostalCode>14000</PostalCode>
<City>MEXICO</City>
<State>D.F.</State>
<TelephoneNumber>5325-9900</TelephoneNumber>
<Country>Mexico</Country>
<Store id="1037">
       <Name>Anthony Servito</Name>
       <TaxId>AIA941123Q99</TaxId>
       <Street>DIVISION DEL NORTE</Street>
       <StreetNumber>1042</StreetNumber>
       <ApartmentNumber />
       <Reference />
       <Neighborhood>COL. VERTIZ NARVARTE</Neighborhood>
       <County>DELEG. BENITO JUAREZ</County>
       <PostalCode>03600</PostalCode>
       <City>MEXICO</City>
       <State>D.F.</State>
       <TelephoneNumber></TelephoneNumber>
       <Country>Mexico</Country>
</Store>
<Store id="1001">
       <Name>Aurora Racho</Name>
       <TaxId>AIA941123Q99</TaxId>
       <Street>HUMBOLT</Street>
       <StreetNumber>26</StreetNumber>
       <ApartmentNumber />
       <Reference />
       <Neighborhood>COL. CENTRO</Neighborhood>
       <County>DELEG. CUAUHTEMOC</County>
       <PostalCode>06040</PostalCode>
       <City>MEXICO</City>
       <State>D.F.</State>
       <TelephoneNumber></TelephoneNumber>
```

```
                    <Country>Mexico</Country>
              </Store>
…
</Company>
```

5.6 EInvoice Version Properties File

The EInvoice `version.properties` file identifies the version of the EInvoice application installation or upgrade. This file is loaded when the application is started. After a successful load, the version is logged in the EInvoice logs.

The version.properties file is stored under the **`<SAP POS Returns Authorization installation folder>`**`\EInvoice\EInvoiceConfig` (for example, `C:\Program Files\SAP\ReturnsAuthorization\EInvoice \EInvoiceConfig`).

Note

The `version.properties` file is classified as read-only; it is not intended to be manually updated at any time.

5.7 Configuring the EInvoice Receipt

There are two main configuration files that are used by EInvoice Receipt:

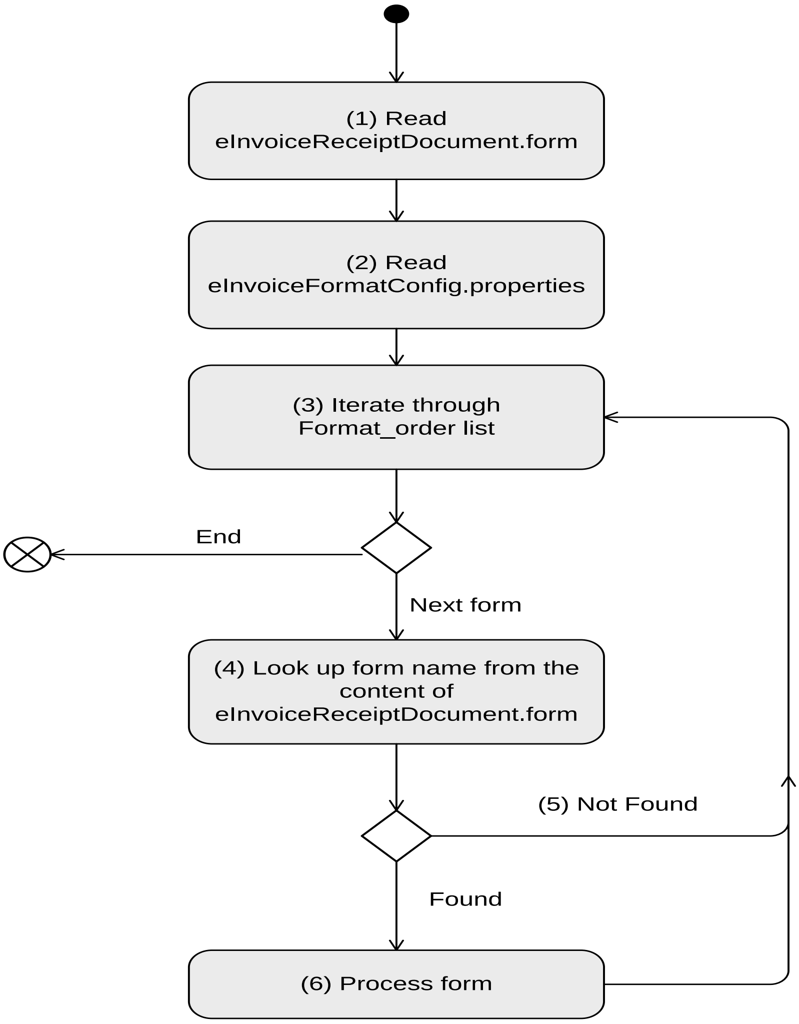
`eInvoiceFormatConfig.properties`

`eInvoiceReceiptDocument.form`

Note

Both configuration files are located under **`<SAP POS Returns Authorization installation folder>`**`\EInvoice\EInvoiceConfig` (the default location for this folder is `C:\Program Files\SAP\ReturnsAuthorization\EInvoice\EInvoiceConfig`).

EInvoice Receipt Flow


(1) Read eInvoiceReceiptDocument.form
(2) Read eInvoiceFormatConfig.properties
(3) Iterate through Format_order list
End
Next form
(4) Look up form name from the content of eInvoiceReceiptDocument.form
(5) Not Found
Found
(6) Process form

An EInvoice receipt is constructed or processed by the EInvoice Receipt component as follows:

1. Reads and caches the content of the `eInvoiceReceiptDocument.form` configuration file. This occurs only once during the EInvoice application start up.
2. Reads and caches the list of forms configured in the `format_order` property of the `eInvoiceFormatConfig.properties` file. This occurs only once during the EInvoice application start up.
3. Processes the form by iterating through the list of forms according to the order configured in the `format_order` property.
4. The current form is looked up from the cached content of the `eInvoiceReceiptDocument.form` configuration file.
5. If the current form is not found, the process returns to step 3 with the next form listed in the `format_order` property.
6. If the current form is found, the form is processed according to the logic and contents of the form. The result is appended to the EInvoice receipt being constructed.
7. The process returns to step 3 with the next form listed in the `format_order` property.
8. The process ends when the iteration through all of the forms has been completed.

5.8 Configuring SAP Point-of-Sale

For the SAP POS application to work with the EInvoice application, you must make the following configuration changes using the Configurator application of SAP POS:

5.9 Configuring Browser Entries

1. From the *POS Configurations* window, on the *Transaction Functions* tab, click *POS Browser*.

   The *POS Browser Search* window appears. This window displays the browser IDs that are currently defined. Each browser entry is associated with an EInvoice operation – *Quick Einvoice*, *Print EInvoice*, *Reprint EInvoice*, and *Cancel EInvoice*.

2. To configure a browser entry, double-click the desired line. Alternatively, click the desired line and then choose *Open*.

   The *POS Browser Maintenance* window appears.

3. For the selected browser entry, in the URL field, specify the following query string:

   Quick Einvoice

   *http://***<IP Address>***:9080/EInvoiceWebApp/pages/index.jsp?eifunction=print&store=@@store&register=@@register&company=***<1>***&cashier=@@cashier&trxnumber=@@trxnumber&businessdate=@@businessdate*

   Print Einvoice

   *http://***<IP Address>***:9080/EInvoiceWebApp/pages/index.jsp?eifunction=print&store=@@store&register=@@register&company=***<1>***&cashier=@@cashier*

   Reprint Einvoice

   *http://***<IP Address>***:9080/EInvoiceWebApp/pages/index.jsp?eifunction=reprint&store=@@store&register=@@register&company=***<1>***&cashier=@@cashier*

   Cancel Einvoice

   *http://***<IP Address>***:9080/EInvoiceWebApp/pages/index.jsp?eifunction=cancel&store=@@store&register=@@register&company=***<1>***&cashier=@@cashier*

   The variables `@@store`, `@@register`, and `@@cashier` are expanded automatically with the actual values for store, register, cashier, transaction number, and business date by the POS before invoking the URL. Where the **`<IP Address>`** is the IP address where SAP Returns Authorization is installed and deployed. **`<1>`** is the company ID.

   1. If the browser entry requires the printing of an EInvoice (*EINVOICE PRINT*, *EINVOICE REPRINT*, *QUICK INVOICE*), select the *Allow Receipt Printing* checkbox.

      The selection of this checkbox tells SAP POS that the response from the browser contains an embedded EInvoice. The EInvoice will be printed on the receipt printer and entered in the journal.To save the changes, choose *Save*.

   2. To configure the next browser entry, click the forward arrow at the bottom of the screen and repeat the above steps

   3. When all browser entries are configured, click *Close*.

4. On the POS Configurations window, click *Convert*. A message displays indicating that the conversion is done.

5.9.1 Associating a Browser Entry with a Manager Code

You must associate a browser entry with a manager code; for example, browser ID 1 is associated with manager code 751, browser ID 2 is associated with manager code 752, and so on, up to browser ID 9, manager code 759.

Before you begin, ensure that you have configured the relevant browser entry. For more information, see section *5.9*, *Configuring Browser Entries*.

1. On the *POS Configuration* window, choose the *Code Maintenance* tab and select *POS Mgr Functions*.
2. The *POS/Manager Function List Maintenance* window appears.
3. Add/Update the appropriate manager code for each browser entry.
4. To save the changes, choose *Save*.
5. When all manager code entries are configured, select *Close*.
6. On the *POS Configurations* window, choose *Convert*. A message displays indicating that the conversion is complete.

5.9.2 Associating a Browser Entry with a Menu

You must associate a browser entry with a menu.

Before you begin, ensure that you have configured the relevant browser entry and the relevant manager code. For more information, see sections *5.9*, *Configuring Browser Entries* and *5.9.1*, *Associating a Browser Entry with a Manager Code*.

1. Add an Einvoice custom menu at the Items level. From the POS Configurations window, on the *Terminal Functions* tab, click the *Custom Menu(Items)* icon.
2. On the *Custom Menu Names* window, choose *New*. Enter the *ID*, and the *Description*.
3. Select the newly added custom menu, then select *Open*.
4. On the *Custom Menu Maintenance* window, choose *Add Option*.
5. Choose *Function* and select *Mgr/POS Function*.
6. On the *Mgr/POS Function (Drag and Drop)* window, choose *Mgr Function* (*Quick Einvoice*, *Print EInvoice*, *Reprint EInvoice*, and *Cancel Einvoice*.)
7. Repeat steps 6 and 7 until you add all of the options required.
8. Select *Close* to close the *MGR/POS Function* window.
9. Choose *Save* in the *Custom Menu Maintenance* window.
10. Choose *Save* in the *Custom Menu Names* window.
11. On the *POS Configurations* window, select *Convert*. A message displays indicating that the conversion is done.

A cashier can now use the EInvoice application by either:

- Executing the associated manager code (for example if printing an EInvoice is configured with browser ID 1, running manager code 751 invokes browser entry 1, printing an EInvoice)

- Selecting the relevant function key on the Main menu

5.9.3 Maintaining TLog Entry

For SAP POS to produce the correct TLogs, you must enable a setting in the Configurator.

1. On the *POS Configurations* main window, click the *Terminal* tab.
2. On the *Terminal* window, click *Register Parameters*.
3. On the *Register Parameter Search* window, select *Region*, and choose *Open*.
4. On the *Register Parameters Maintenance* window, choose the *T_logs* tab and click the *Add Itemized Tax Record* checkbox.
5. To save this change and close the transaction, click *Save* and then *Close*.

5.9.4 Changing the POS.INI File

To disallow the post-voiding of a transaction for which an EInvoice has been generated and printed, in the `POS.INI` file, set the `CHECKVOIDFORINVOICEDTXN` parameter to `Y`.

6 Tools and Scripts

6.1 EInvoice Recovery Tool

If there is a failure in persisting generated EInvoices to the EI database, the system administrators can use the recovery tool to manually recover failed EInvoice objects to the EI database from the folder where they are dumped.

6.1.1 Recovery Tools Files

The recovery tool files are included in the SAP Returns Authorization’s EInvoice installation as the `EInvoiceRecoveryTool` folder. However, the folder can be put anywhere, that is, machines other than the SAP Returns Authorization server, as long as the IBM WebSphere jar files and the necessary fail and success folders are available on that machine.

The `EInvoiceRecoveryTool` folder contains the following files:

- `recoverytool.properties`
- `recoverytool.bat`
- `lib folder`, which contains `EInvoiceCore.jar` and `EInvoiceEJBClient.jar`
- classes folder, which contains recovery tool class files

Note

The IBM WebSphere jar files must be in the `WebSphere\AppServer\lib` library in the classpath. Therefore, the `WAS_HOME` environment variable must be set to point to `WebSphere\AppServer`; for example, `C:\Program Files\IBM\WebSphere\AppServer`.

The `recoverytool.properties` file has the following entries:

Parameter	Description
`Ip`	Server IP address: where the EInvoice application is running.
`Port`	Port number.
`Jndiname`	Java Naming and Directory Interface (JNDI) name.
`exceptionDumpPath`	Directory path where the failed invoices are located.
`successPath`	Directory path where the successfully recovered invoices are located.

Example

```
ip=10.6.72.90

port=2809

jndiname=EInvoiceService
```

```
exceptionDumpPath=C:\\FailedInvoices
successPath=C:\\FailedInvoices\\Success
```

6.1.2 Running the Recovery Tool

To invoke the recovery tool, the system administrator runs `recoverytool.bat`. Every file in the folder specified in `recoverytool.properties` (`C:\\FailedInvoices` in the example above) is sent to the server to be saved in the EInvoice database.

If the recovery process succeeds, the files of the successfully recovered EInvoices are moved to the folder specified in `recoverytool.properties` (`C:\\FailedInvoices\\Success` in the example above). Otherwise, the files remain in the failed invoices folder.

6.2 EInvoice Data Purge Job

The process to create the database tables also creates a job, called `PurgeDatabase_EI`, that permanently deletes all EInvoice entries that exceed the number of retention days specified in the SAP Returns Authorization Web Console.

The purge job is scheduled to run every day at 1:00 a.m. You can configure the `PurgeDatabase_EI` job in Microsoft SQL Server Management Studio.

6.2.1 Configuring the Data Purge Job

1. Open Microsoft SQL Server Management Studio.
2. Choose *SQL Server Agent* → *Jobs* → *PurgeDatabase_EI*.
3. Right-click *PurgeDatabase_EI* and select *Properties*.

   The *Job Schedule Properties* window appears. Here, you can configure many aspects of the data purge job, such as notifications and a schedule.

6.2.2 Configuring the Data Retention Period

EInvoice entries are purged daily when they exceed the number of retention days configured in the SAP Returns Authorization Web Console. Only EInvoice data is purged; customer data is not purged.

1. Start the SAP Returns Authorization Web Console and log on.
2. Choose *Administration* → *Global Parameters*.
3. Click *Edit* and change the value of *Data Retention Days* as required.

Note

The *Data Retention Days* value is used not only for the data purge job, but also in SAP Returns Authorization.

7 Web Service Interface

This section describes the Web Service as defined by SAP. For each Web Service, the responsible party of the client and server will be indicated.

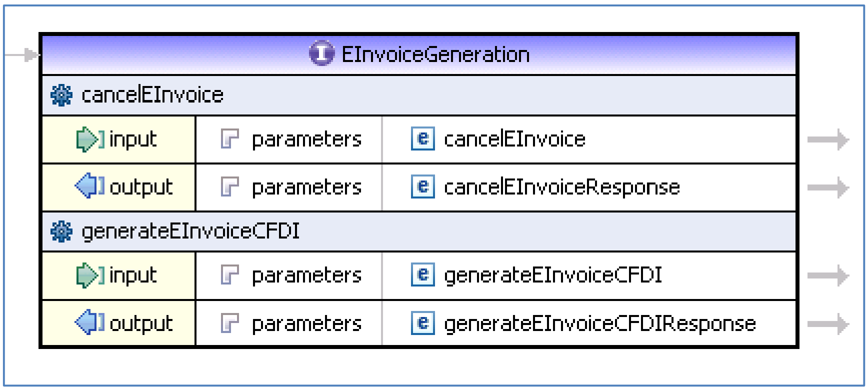
There are three types of interfaces the SAP EInvoice has defined and provided.

1. `EInvoiceGeneration`
2. `EInvoiceMgmt`
3. `CustomerMgmt`

7.1 EInvoiceGeneration

This server-side Web Service interface is implemented by a third-party vendor or your in-house development. It is used for generating and cancelling of electronic invoices.

The following diagram is a high-level view of the EInvoice Generation Web Service interface. The third party vendor or your in-house development must abide by the rules defined by the interface.



EInvoiceGeneration		
cancelEInvoice		
input	parameters	cancelEInvoice
output	parameters	cancelEInvoiceResponse
generateEInvoiceCFDI		
input	parameters	generateEInvoiceCFDI
output	parameters	generateEInvoiceCFDIResponse

7.1.1 generateEInvoiceCFDI

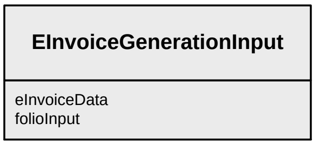
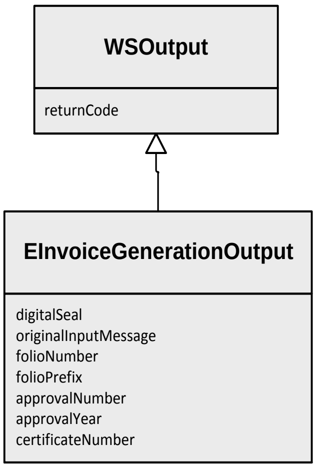
SAP's EInvoice application calls the `generateEInvoiceCFDI` method to generate an EInvoice by sending the EInvoice data to the Third Party Web Service Façade Server, and delegating to the Third Party electronic invoicing application(s). This method provides both the folio input information for the Third Party electronic invoicing application(s) (see the folioInput parameter in the table below), which contains the required parameters and the EInvoice data:

1. `companyId`
2. `storeId`
3. `documentType`

4. `id1` (unique identifier)
5. `id2` (unique identifier)

The output of this call method (see EInvoiceGenerationOutput) contains both the result from the Third Party Web Service Façade Server and the digital seal and original input message from the Third Party electronic invoicing application(s).

This method has the following signature:

Input	
Output	 WSOutput returnCode EInvoiceGenerationOutput digitalSeal originalInputMessage folioNumber folioPrefix approvalNumber approvalYear certificateNumber

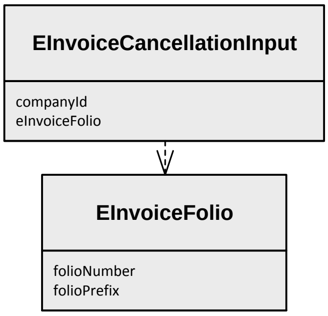
The return codes used for this method are:

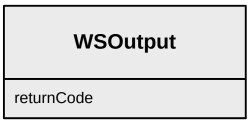
1. `Success`
2. `Internal` error
3. `EInvoice already created`

7.1.2 cancelEInvoice

SAP's EInvoice application calls this method to cancel an EInvoice by sending the EInvoice data to the Third Party Web Service Façade Server, and in turn delegates to the Third Party electronic invoicing application(s).

This method has the following signature:

Input	 EInvoiceCancellationInput companyId eInvoiceFolio EInvoiceFolio folioNumber folioPrefix

Output	

The return codes that are used for this method are:

4. `Success`
5. `EInvoice does not exist`
6. `Internal error`
7. `eInvoice already cancelled`

For this server-side Web Service, SAP implements the client side and the third party vendor implements the server side.

Below is a WSDL that defines the EInvoice Generation web service interface:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://generation.ws.einvoice.pos.sap.com"
xmlns:impl="http://generation.ws.einvoice.pos.sap.com"
xmlns:intf="http://generation.ws.einvoice.pos.sap.com"
xmlns:tns2="http://parameter.ws.einvoice.pos.sap.com"
xmlns:tns3="http://common.ws.einvoice.pos.sap.com"
xmlns:tns4="http://common.model.einvoice.pos.sap.com"
xmlns:tns5="http://external.model.einvoice.pos.sap.com"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsi="http://ws-
i.org/profiles/basic/1.1/xsd" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <wsdl:types>
  <schema targetNamespace="http://parameter.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://external.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.ws.einvoice.pos.sap.com"/>
   <complexType name="EInvoiceKeyInput">
    <complexContent>
     <extension base="tns3:WSInput">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="EInvoiceFolio" nillable="true" type="tns4:EInvoiceFolio"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="EInvoiceInput">
    <complexContent>
     <extension base="tns3:WSInput">
```

```
      <sequence>

       <element name="EInvoice" nillable="true" type="tns5:EInvoice"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="EInvoiceOutput">
    <complexContent>
     <extension base="tns3:WSOutput">
      <sequence>
       <element name="EInvoice" nillable="true" type="tns5:EInvoice"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="WSInput">
    <sequence>
     <element name="sourceSystemCode" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="WSOutput">
    <sequence>
     <element name="returnCode" type="xsd:int"/>
     <element name="returnMessage" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="EInvoiceFolio">
    <sequence>
     <element name="folioNumber" type="xsd:int"/>
     <element name="folioPrefix" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseEInvoice">
    <sequence>
     <element name="invoiceDate" nillable="true" type="xsd:dateTime"/>
     <element name="cancelled" type="xsd:boolean"/>
     <element name="cancelDate" nillable="true" type="xsd:dateTime"/>
     <element name="originalInputMessage" nillable="true" type="xsd:string"/>
     <element name="digitalSeal" nillable="true" type="xsd:string"/>
     <element name="approvalNumber" type="xsd:int"/>
     <element name="approvalYear" type="xsd:int"/>
     <element name="certificateNumber" nillable="true" type="xsd:string"/>
     <element name="trxStoreId" type="xsd:int"/>
     <element name="trxRegisterId" type="xsd:int"/>
     <element name="trxNumber" type="xsd:long"/>
     <element name="trxBusinessDate" nillable="true" type="xsd:dateTime"/>
     <element name="trxCashierId" type="xsd:long"/>
     <element name="amountOnText" nillable="true" type="xsd:string"/>
     <element name="trxOriginCode" nillable="true" type="xsd:string"/>
     <element name="xmlPath" nillable="true" type="xsd:string"/>
     <element name="pdfPath" nillable="true" type="xsd:string"/>
     <element name="documentType" nillable="true" type="xsd:string"/>
     <element name="insertBy" nillable="true" type="xsd:string"/>
     <element name="cancelBy" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseCustomer">
    <complexContent>
     <extension base="tns4:BaseUser">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="insertDate" nillable="true" type="xsd:dateTime"/>
       <element name="lastChangedDate" nillable="true" type="xsd:dateTime"/>
       <element name="insertBy" nillable="true" type="xsd:string"/>
       <element name="lastChangedBy" nillable="true" type="xsd:string"/>
       <element name="customerType" nillable="true" type="xsd:string"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseUser">
    <sequence>
     <element name="taxId" nillable="true" type="xsd:string"/>
     <element name="name1" nillable="true" type="xsd:string"/>
     <element name="name2" nillable="true" type="xsd:string"/>
```

```
    </sequence>
   </complexType>
   <complexType name="Address">
    <sequence>
     <element name="streetName" nillable="true" type="xsd:string"/>
     <element name="city" nillable="true" type="xsd:string"/>
     <element name="county" nillable="true" type="xsd:string"/>
     <element name="state" nillable="true" type="xsd:string"/>
     <element name="country" nillable="true" type="xsd:string"/>
     <element name="postalCode" nillable="true" type="xsd:string"/>
     <element name="streetNumber" nillable="true" type="xsd:string"/>
     <element name="apartmentNumber" nillable="true" type="xsd:string"/>
     <element name="neighborhood" nillable="true" type="xsd:string"/>
     <element name="reference" nillable="true" type="xsd:string"/>
     <element name="telephoneNumber" nillable="true" type="xsd:string"/>
     <element name="email" nillable="true" type="xsd:string"/>
     <element name="sequenceNumber" type="xsd:long"/>
    </sequence>
   </complexType>

       <element name="EInvoice" nillable="true" type="tns5:EInvoice"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="EInvoiceOutput">
    <complexContent>
     <extension base="ts3:WSOutput">
      <sequence>
       <element name="EInvoice" nillable="true" type="tns5:EInvoice"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="WSInput">
    <sequence>
     <element name="sourceSystemCode" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="WSOutput">
    <sequence>
     <element name="returnCode" type="xsd:int"/>
     <element name="returnMessage" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="EInvoiceFolio">
    <sequence>
     <element name="folioNumber" type="xsd:int"/>
     <element name="folioPrefix" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseEInvoice">
    <sequence>
     <element name="invoiceDate" nillable="true" type="xsd:dateTime"/>
     <element name="cancelled" type="xsd:boolean"/>
     <element name="cancelDate" nillable="true" type="xsd:dateTime"/>
     <element name="originalInputMessage" nillable="true" type="xsd:string"/>
     <element name="digitalSeal" nillable="true" type="xsd:string"/>
     <element name="approvalNumber" type="xsd:int"/>
     <element name="approvalYear" type="xsd:int"/>
     <element name="certificateNumber" nillable="true" type="xsd:string"/>
     <element name="trxStoreId" type="xsd:int"/>
     <element name="trxRegisterId" type="xsd:int"/>
     <element name="trxNumber" type="xsd:long"/>
     <element name="trxBusinessDate" nillable="true" type="xsd:dateTime"/>
     <element name="trxCashierId" type="xsd:long"/>
     <element name="amountOnText" nillable="true" type="xsd:string"/>
     <element name="trxOriginCode" nillable="true" type="xsd:string"/>
     <element name="xmlPath" nillable="true" type="xsd:string"/>
     <element name="pdfPath" nillable="true" type="xsd:string"/>
     <element name="documentType" nillable="true" type="xsd:string"/>
     <element name="insertBy" nillable="true" type="xsd:string"/>
     <element name="cancelBy" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseCustomer">
```

```
 <complexContent>
  <extension base="tns4:BaseUser">
   <sequence>
    <element name="companyId" type="xsd:int"/>
    <element name="insertDate" nillable="true" type="xsd:dateTime"/>
    <element name="lastChangedDate" nillable="true" type="xsd:dateTime"/>
    <element name="insertBy" nillable="true" type="xsd:string"/>
    <element name="lastChangedBy" nillable="true" type="xsd:string"/>
    <element name="customerType" nillable="true" type="xsd:string"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
<complexType abstract="true" name="BaseUser">
 <sequence>
  <element name="taxId" nillable="true" type="xsd:string"/>
  <element name="name1" nillable="true" type="xsd:string"/>
  <element name="name2" nillable="true" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="Address">
 <sequence>
  <element name="streetName" nillable="true" type="xsd:string"/>
  <element name="city" nillable="true" type="xsd:string"/>
  <element name="county" nillable="true" type="xsd:string"/>
  <element name="state" nillable="true" type="xsd:string"/>
  <element name="country" nillable="true" type="xsd:string"/>
  <element name="postalCode" nillable="true" type="xsd:string"/>
  <element name="streetNumber" nillable="true" type="xsd:string"/>
  <element name="apartmentNumber" nillable="true" type="xsd:string"/>
  <element name="neighborhood" nillable="true" type="xsd:string"/>
  <element name="reference" nillable="true" type="xsd:string"/>
  <element name="telephoneNumber" nillable="true" type="xsd:string"/>
  <element name="email" nillable="true" type="xsd:string"/>
  <element name="sequenceNumber" type="xsd:long"/>
 </sequence>
</complexType>

<complexType abstract="true" name="BaseStore">
 <complexContent>
  <extension base="tns4:BaseUser">
```

```
      <sequence>
       <element name="storeId" type="xsd:int"/>
       <element name="address" nillable="true" type="tns4:Address"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseCompany">
    <complexContent>
     <extension base="tns4:BaseUser">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="address" nillable="true" type="tns4:Address"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseLineItem">
    <sequence>
     <element name="sku" nillable="true" type="xsd:string"/>
     <element name="returned" type="xsd:boolean"/>
     <element name="department" nillable="true" type="xsd:string"/>
     <element name="description" nillable="true" type="xsd:string"/>
     <element name="qty" type="xsd:double"/>
     <element name="uniqueId" nillable="true" type="xsd:string"/>
     <element name="weighted" type="xsd:boolean"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseTaxInformation">
    <sequence>
     <element name="id" nillable="true" type="xsd:string"/>
     <element name="taxDescription" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://generation.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://external.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.ws.einvoice.pos.sap.com"/>
   <import namespace="http://parameter.ws.einvoice.pos.sap.com"/>
   <element name="cancelEInvoiceResponse">
    <complexType>
     <sequence>
      <element name="cancelEInvoiceReturn" nillable="true" type="tns3:WSOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="generateEInvoice">
    <complexType>
     <sequence>
      <element name="eInvoiceInput" nillable="true" type="tns2:EInvoiceInput"/>
     </sequence>
    </complexType>
   </element>
   <element name="generateEInvoiceResponse">
    <complexType>
     <sequence>
      <element name="generateEInvoiceReturn" nillable="true"
type="tns2:EInvoiceOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="cancelEInvoice">
    <complexType>
     <sequence>
      <element name="eInvoiceKeyInput" nillable="true" type="tns2:EInvoiceKeyInput"/>
     </sequence>
    </complexType>
   </element>
   <complexType name="ArrayOf_tns4_nillable_Address">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="Address" nillable="true"
type="tns4:Address"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOf_tns5_nillable_TaxInformation">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="TaxInformation"
nillable="true" type="tns5:TaxInformation"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOf_tns5_nillable_LineItem">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="LineItem" nillable="true"
type="tns5:LineItem"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOf_tns5_nillable_TaxTypeSubTotal">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="TaxTypeSubTotal"
nillable="true" type="tns5:TaxTypeSubTotal"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://external.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://generation.ws.einvoice.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <complexType name="EInvoice">
    <complexContent>
     <extension base="tns4:BaseEInvoice">
      <sequence>

       <element name="taxAmount" type="xsd:double"/>
       <element name="totalDiscountAmount" type="xsd:double"/>
       <element name="netAmount" type="xsd:double"/>
       <element name="folio" nillable="true" type="tns4:EInvoiceFolio"/>
       <element name="customer" nillable="true" type="tns5:Customer"/>
       <element name="invoiceStoreInfo" nillable="true" type="tns5:Store"/>
       <element name="EInvoiceItems" nillable="true"
type="impl:ArrayOf_tns5_nillable_LineItem"/>
       <element name="taxTypeSubTotalList" nillable="true"
type="impl:ArrayOf_tns5_nillable_TaxTypeSubTotal"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="Customer">
    <complexContent>
     <extension base="tns4:BaseCustomer">
      <sequence>
       <element name="addressList" nillable="true"
type="impl:ArrayOf_tns4_nillable_Address"/>
```

```
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="Store">
    <complexContent>
     <extension base="tns4:BaseStore">
      <sequence>
       <element name="companyInfo" nillable="true" type="tns5:Company"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="Company">
    <complexContent>
     <extension base="tns4:BaseCompany">
      <sequence/>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="LineItem">
    <complexContent>
     <extension base="tns4:BaseLineItem">
      <sequence>
       <element name="grossPrice" type="xsd:double"/>
       <element name="netPrice" type="xsd:double"/>
       <element name="unitPrice" type="xsd:double"/>
       <element name="taxInfoList" nillable="true"
type="impl:ArrayOf_tns5_nillable_TaxInformation"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="TaxInformation">
    <complexContent>
     <extension base="tns4:BaseTaxInformation">
      <sequence>
       <element name="taxRate" type="xsd:double"/>
       <element name="taxAmount" type="xsd:double"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="TaxTypeSubTotal">
    <sequence>
     <element name="subTotalAmount" type="xsd:double"/>
     <element name="taxInformation" nillable="true" type="tns5:TaxInformation"/>
    </sequence>
   </complexType>
  </schema>
 </wsdl:types>

   <wsdl:message name="generateEInvoiceRequest">
     <wsdl:part element="impl:generateEInvoice" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="cancelEInvoiceRequest">
     <wsdl:part element="impl:cancelEInvoice" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="generateEInvoiceResponse">
     <wsdl:part element="impl:generateEInvoiceResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="cancelEInvoiceResponse">
     <wsdl:part element="impl:cancelEInvoiceResponse" name="parameters"/>
  </wsdl:message>

   <wsdl:portType name="EInvoiceGeneration">
     <wsdl:operation name="cancelEInvoice">
       <wsdl:input message="impl:cancelEInvoiceRequest" name="cancelEInvoiceRequest"/>

       <wsdl:output message="impl:cancelEInvoiceResponse"
name="cancelEInvoiceResponse"/>
    </wsdl:operation>

     <wsdl:operation name="generateEInvoice">
       <wsdl:input message="impl:generateEInvoiceRequest"
name="generateEInvoiceRequest"/>
```

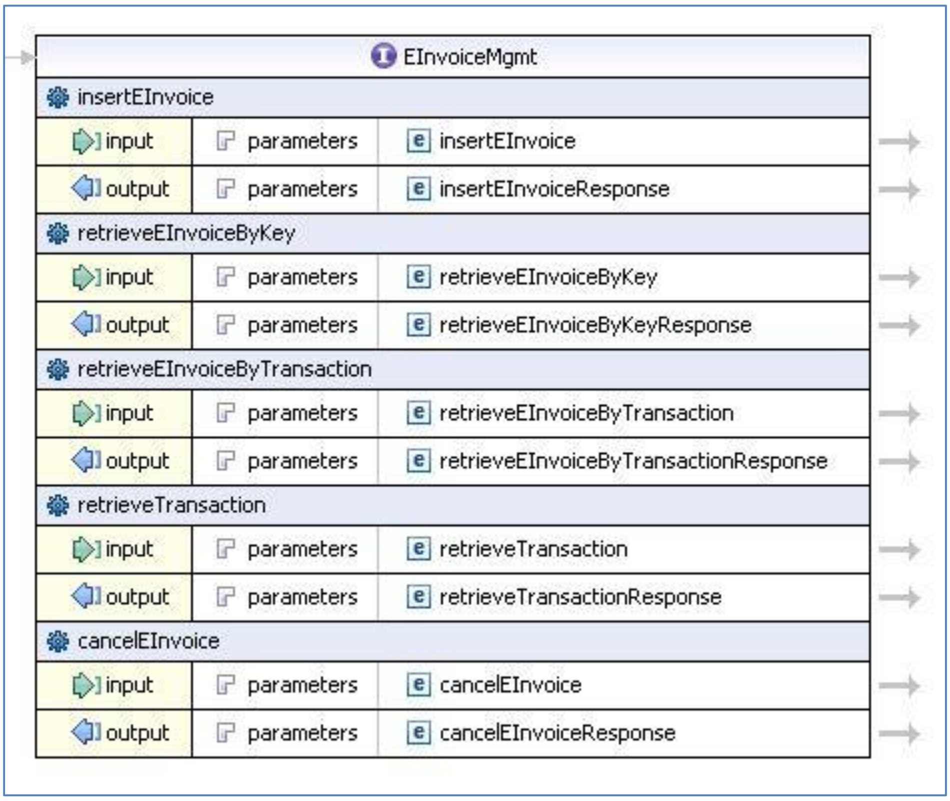
```
       <wsdl:output message="impl:generateEInvoiceResponse"
name="generateEInvoiceResponse"/>

    </wsdl:operation>
  </wsdl:portType>

   <wsdl:binding name="EInvoiceGenerationSoapBinding" type="impl:EInvoiceGeneration">
     <wsdlsoap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>

     <wsdl:operation name="cancelEInvoice">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="cancelEInvoiceRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="cancelEInvoiceResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

     <wsdl:operation name="generateEInvoice">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="generateEInvoiceRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="generateEInvoiceResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

  </wsdl:binding>
```

```
   <wsdl:service name="EInvoiceGenerationService">

     <wsdl:port binding="impl:EInvoiceGenerationSoapBinding"
name="EInvoiceGeneration">

       <wsdlsoap:address
location="http://localhost:9080/EInvoiceWebServices/services/EInvoiceGeneration"/>


    </wsdl:port>


  </wsdl:service>



</wsdl:definitions>
```

7.2 EInvoice Mgmt Interface

SAP EInvoice implemented this client-side web service interface to allow vendor's third-party application to insert, retrieve and cancel EInvoice into and from the SAP EInvoice database. SAP implemented the server-side and the retailer/vendor must implement the client-side.

The following diagram is a high-level view of the EInvoice Mgmt web service interface. The third party vendor should abide by the rules defined by the interface.

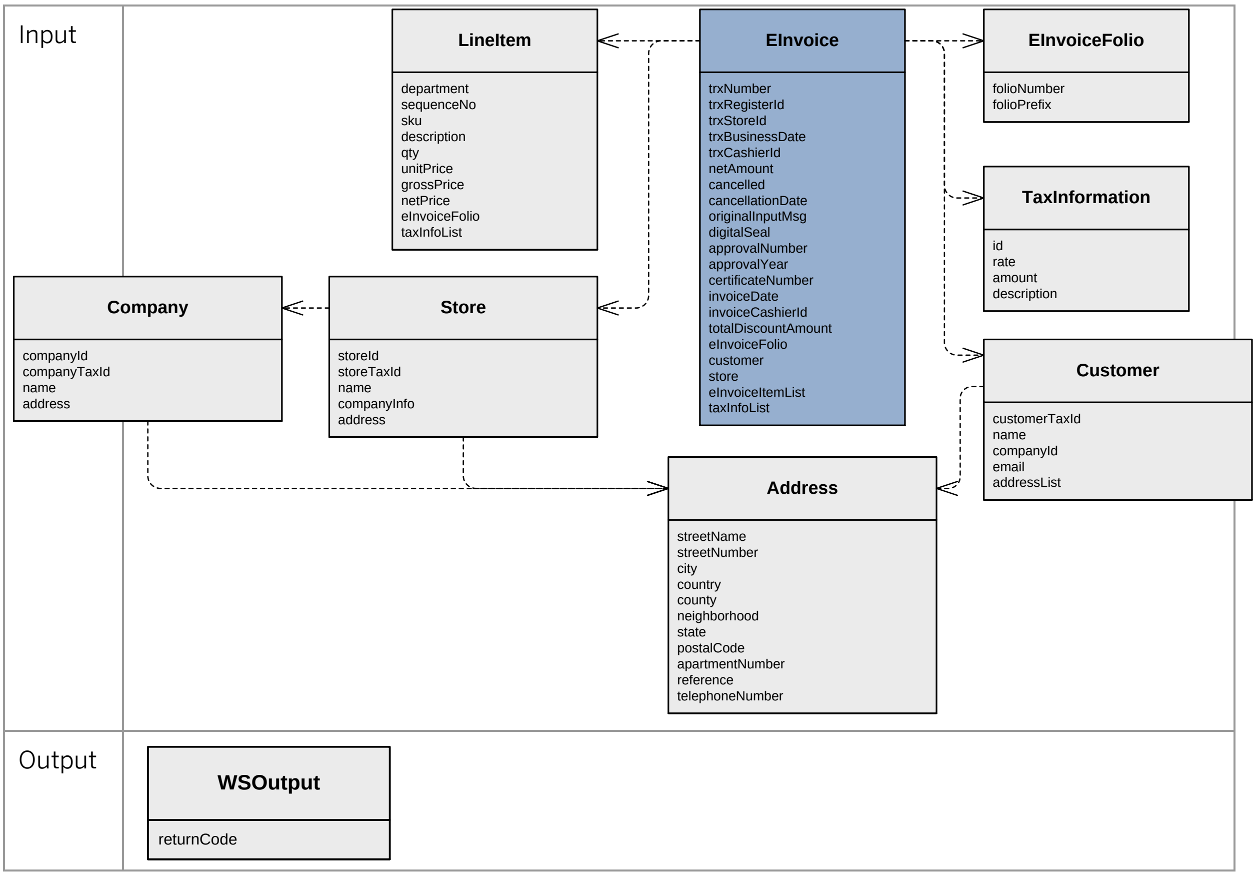


EInvoiceMgmt		
insertEInvoice		
input	parameters	insertEInvoice
output	parameters	insertEInvoiceResponse
retrieveEInvoiceByKey		
input	parameters	retrieveEInvoiceByKey
output	parameters	retrieveEInvoiceByKeyResponse
retrieveEInvoiceByTransaction		
input	parameters	retrieveEInvoiceByTransaction
output	parameters	retrieveEInvoiceByTransactionResponse
retrieveTransaction		
input	parameters	retrieveTransaction
output	parameters	retrieveTransactionResponse
cancelEInvoice		
input	parameters	cancelEInvoice
output	parameters	cancelEInvoiceResponse

7.2.1 insertEInvoice

The retailer/vendor's application calls this method to create an EInvoice in SAP's EInvoice database.

This method has the following signature:



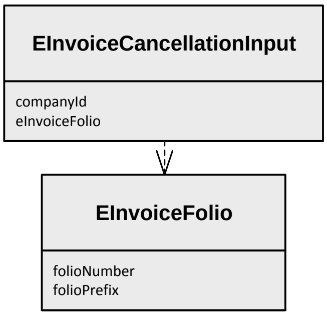
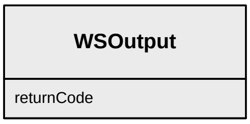
The return codes used for this method are:

1. `Success`
2. `EInvoice already exists`
3. `Internal error`

7.2.2 cancelEInvoice

The retailer/vendor's application calls this method to set the status of an EInvoice record to Cancelled in SAP's EInvoice database.

This method has the following signature:

Input	 EInvoiceCancellationInput companyId eInvoiceFolio EInvoiceFolio folioNumber folioPrefix
Output	

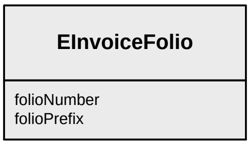
The return codes used for this method are:

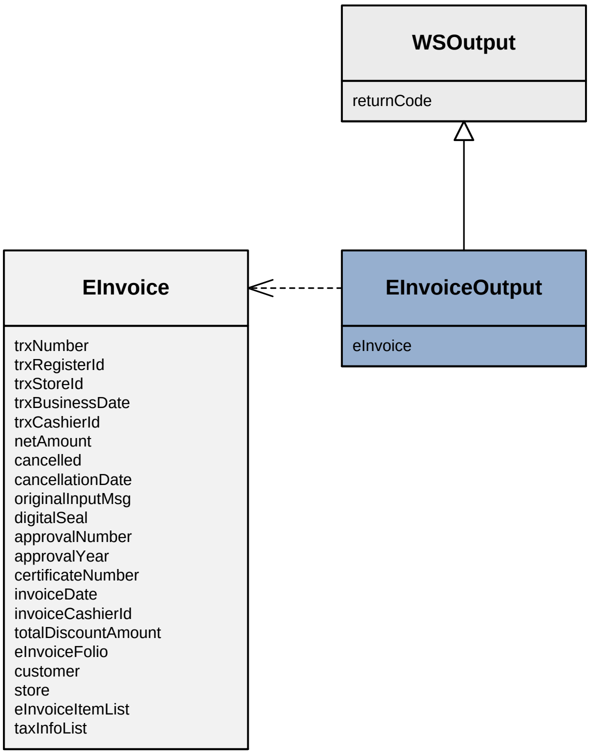
1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

7.2.3 retrieveEInvoiceByFolio

The retailer/vendor's application calls this method to retrieve an EInvoice record by Folio information in SAP's EInvoice database.

This method has the following signature:

Input	

Output	 WSOutput returnCode EInvoice trxNumber trxRegisterId trxStoreId trxBusinessDate trxCashierId netAmount cancelled cancellationDate originalInputMsg digitalSeal approvalNumber approvalYear certificateNumber invoiceDate invoiceCashierId totalDiscountAmount eInvoiceFolio customer store eInvoiceItemList taxInfoList EInvoiceOutput eInvoice

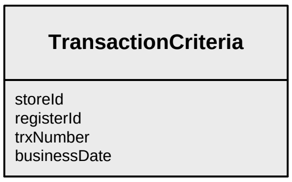
The return codes that are used for this method are:

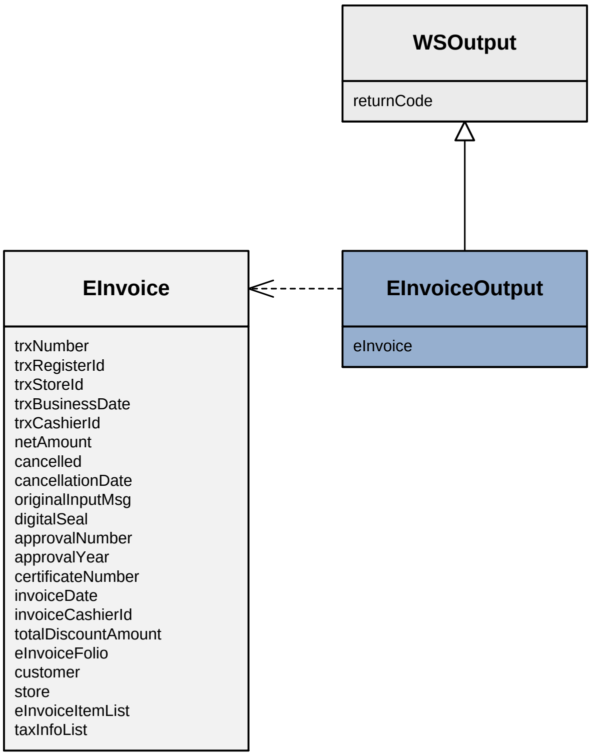
1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

7.2.4 retrieveEInvoiceByTransaction

The retailer/vendor's application shall call this method in order to retrieve an EInvoice record by Transaction information in SAP's EInvoice database.

This method has the following signature:

Input	 TransactionCriteria storeId registerId trxNumber businessDate

Output	

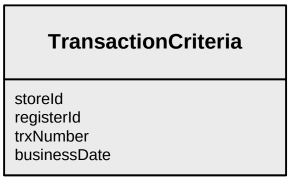
The return codes that are used for this method are:

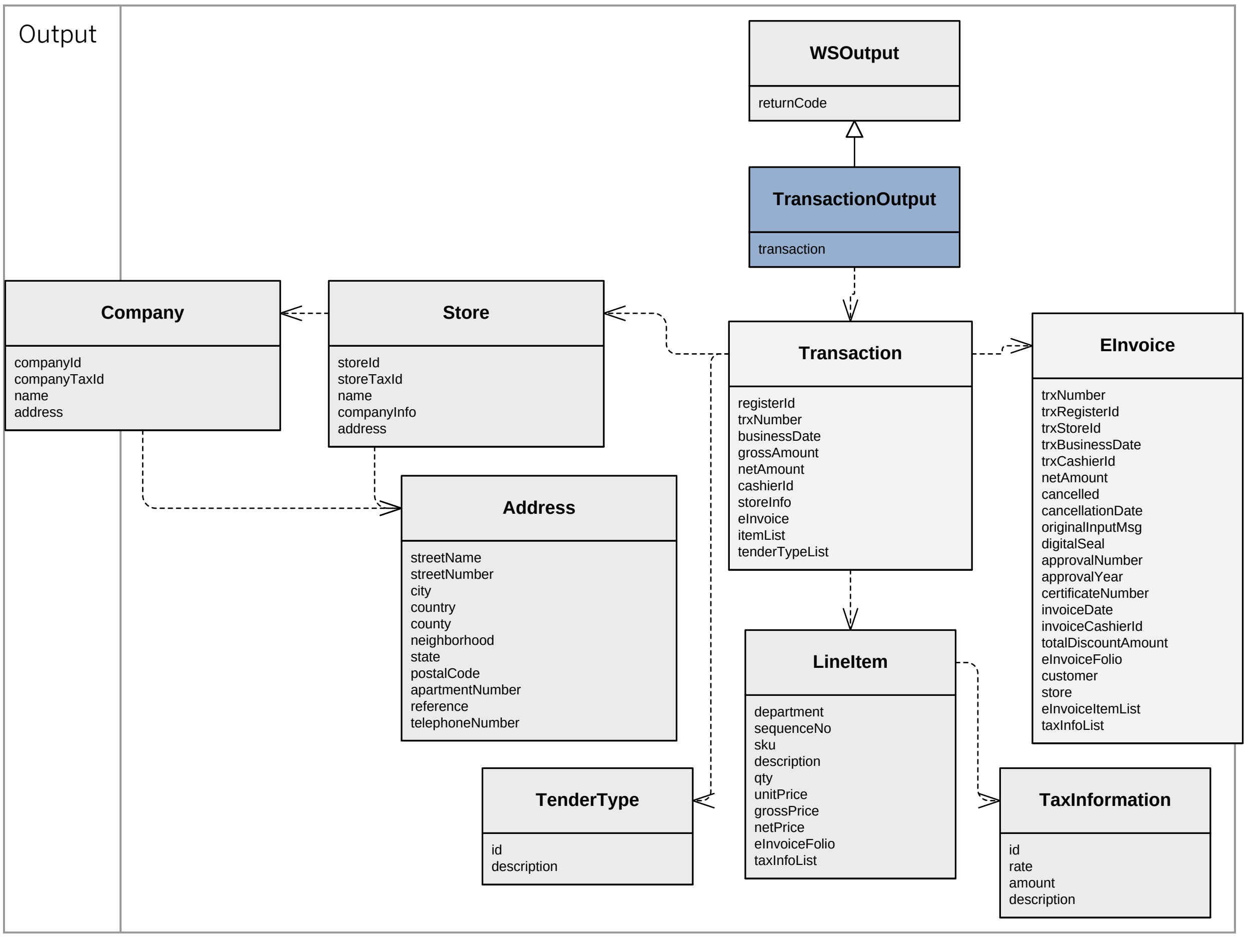
1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

7.2.5 retrieveTransaction

The retailer/vendor's application calls this method to retrieve a transaction record, along with any existing related EInvoices in SAP's transaction/EInvoice databases.

This method has the following signature:

Input	



The return codes that are used for this method are:

1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

Below is a WSDL that defines the EInvoice Mgmt web service interface.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://mgmt.ws.einvoice.pos.sap.com"
xmlns:impl="http://mgmt.ws.einvoice.pos.sap.com"
xmlns:intf="http://mgmt.ws.einvoice.pos.sap.com"
xmlns:tns2="http://parameter.ws.einvoice.pos.sap.com"
xmlns:tns3="http://common.ws.einvoice.pos.sap.com"
xmlns:tns4="http://external.model.einvoice.pos.sap.com"
xmlns:tns5="http://common.model.einvoice.pos.sap.com"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsi="http://ws-
i.org/profiles/basic/1.1/xsd" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <wsdl:types>
  <schema targetNamespace="http://parameter.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
```

```
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <import namespace="http://external.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.ws.einvoice.pos.sap.com"/>
   <complexType name="EInvoiceInput">
    <complexContent>
     <extension base="tns3:WSInput">
      <sequence>
       <element name="EInvoice" nillable="true" type="tns4:EInvoice"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="EInvoiceKeyInput">
    <complexContent>
     <extension base="tns3:WSInput">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="EInvoiceFolio" nillable="true" type="tns5:EInvoiceFolio"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="EInvoiceOutput">
    <complexContent>
     <extension base="tns3:WSOutput">
      <sequence>
       <element name="EInvoice" nillable="true" type="tns4:EInvoice"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>

   <complexType name="TransactionCriteriaInput">
    <complexContent>
     <extension base="tns3:WSInput">
      <sequence>
       <element name="transactionCriteria" nillable="true"
type="tns2:TransactionCriteria"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="TransactionCriteria">
    <sequence>
     <element name="storeId" type="xsd:int"/>
     <element name="registerId" type="xsd:int"/>
     <element name="trxNumber" type="xsd:long"/>
     <element name="businessDate" nillable="true" type="xsd:dateTime"/>
    </sequence>
   </complexType>
   <complexType name="TransactionOutput">
    <complexContent>
     <extension base="tns3:WSOutput">
      <sequence>
       <element name="transaction" nillable="true" type="tns4:Transaction"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="EInvoiceCancellationInput">
    <complexContent>
     <extension base="tns2:EInvoiceKeyInput">
      <sequence>
       <element name="cancelBy" nillable="true" type="xsd:string"/>
       <element name="cancelDate" nillable="true" type="xsd:dateTime"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="WSInput">
    <sequence>
     <element name="sourceSystemCode" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="WSOutput">
    <sequence>
     <element name="returnCode" type="xsd:int"/>
     <element name="returnMessage" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://external.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://mgmt.ws.einvoice.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <complexType name="EInvoice">
    <complexContent>
     <extension base="tns5:BaseEInvoice">
      <sequence>
       <element name="taxAmount" type="xsd:double"/>
       <element name="totalDiscountAmount" type="xsd:double"/>
       <element name="netAmount" type="xsd:double"/>
       <element name="folio" nillable="true" type="tns5:EInvoiceFolio"/>
       <element name="customer" nillable="true" type="tns4:Customer"/>
       <element name="invoiceStoreInfo" nillable="true" type="tns4:Store"/>
       <element name="EInvoiceItems" nillable="true"
type="impl:ArrayOf_tns4_nillable_LineItem"/>
       <element name="taxTypeSubTotalList" nillable="true"
type="impl:ArrayOf_tns4_nillable_TaxTypeSubTotal"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="Customer">
    <complexContent>
     <extension base="tns5:BaseCustomer">
      <sequence>
       <element name="addressList" nillable="true"
type="impl:ArrayOf_tns5_nillable_Address"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="Store">
    <complexContent>
     <extension base="tns5:BaseStore">
      <sequence>
       <element name="companyInfo" nillable="true" type="tns4:Company"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="Company">
    <complexContent>
     <extension base="tns5:BaseCompany">
      <sequence/>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="LineItem">
    <complexContent>
     <extension base="tns5:BaseLineItem">
      <sequence>
       <element name="grossPrice" type="xsd:double"/>
       <element name="netPrice" type="xsd:double"/>
       <element name="unitPrice" type="xsd:double"/>
       <element name="taxInfoList" nillable="true"
type="impl:ArrayOf_tns4_nillable_TaxInformation"/>

      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="TaxInformation">
    <complexContent>
     <extension base="tns5:BaseTaxInformation">
      <sequence>
       <element name="taxRate" type="xsd:double"/>
       <element name="taxAmount" type="xsd:double"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="TaxTypeSubTotal">
    <sequence>
     <element name="subTotalAmount" type="xsd:double"/>
     <element name="taxInformation" nillable="true" type="tns4:TaxInformation"/>
    </sequence>
   </complexType>
   <complexType name="Transaction">
    <complexContent>
```

```
     <extension base="tns5:BaseTransaction">
      <sequence>
       <element name="storeInfo" nillable="true" type="tns4:Store"/>
       <element name="EInvoice" nillable="true" type="tns4:EInvoice"/>
       <element name="itemList" nillable="true"
type="impl:ArrayOf_tns4_nillable_LineItem"/>
       <element name="tenderTypeList" nillable="true"
type="impl:ArrayOf_tns4_nillable_TenderType"/>
       <element name="grossAmount" type="xsd:double"/>
       <element name="netAmount" type="xsd:double"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="TenderType">
    <complexContent>
     <extension base="tns5:BaseTenderType">
      <sequence/>
     </extension>
    </complexContent>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType abstract="true" name="BaseEInvoice">
    <sequence>
     <element name="invoiceDate" nillable="true" type="xsd:dateTime"/>
     <element name="cancelled" type="xsd:boolean"/>
     <element name="cancelDate" nillable="true" type="xsd:dateTime"/>
     <element name="originalInputMessage" nillable="true" type="xsd:string"/>
     <element name="digitalSeal" nillable="true" type="xsd:string"/>
     <element name="approvalNumber" type="xsd:int"/>
     <element name="approvalYear" type="xsd:int"/>
     <element name="certificateNumber" nillable="true" type="xsd:string"/>
     <element name="trxStoreId" type="xsd:int"/>
     <element name="trxRegisterId" type="xsd:int"/>
     <element name="trxNumber" type="xsd:long"/>
     <element name="trxBusinessDate" nillable="true" type="xsd:dateTime"/>
     <element name="trxCashierId" type="xsd:long"/>
     <element name="amountOnText" nillable="true" type="xsd:string"/>
     <element name="trxOriginCode" nillable="true" type="xsd:string"/>
     <element name="xmlPath" nillable="true" type="xsd:string"/>
     <element name="pdfPath" nillable="true" type="xsd:string"/>
     <element name="documentType" nillable="true" type="xsd:string"/>
     <element name="insertBy" nillable="true" type="xsd:string"/>
     <element name="cancelBy" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="EInvoiceFolio">
    <sequence>
     <element name="folioNumber" type="xsd:int"/>
     <element name="folioPrefix" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseCustomer">
    <complexContent>
     <extension base="tns5:BaseUser">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="insertDate" nillable="true" type="xsd:dateTime"/>
       <element name="lastChangedDate" nillable="true" type="xsd:dateTime"/>
       <element name="insertBy" nillable="true" type="xsd:string"/>
       <element name="lastChangedBy" nillable="true" type="xsd:string"/>
       <element name="customerType" nillable="true" type="xsd:string"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseUser">
    <sequence>
     <element name="taxId" nillable="true" type="xsd:string"/>
     <element name="name1" nillable="true" type="xsd:string"/>
     <element name="name2" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>

   <complexType name="Address">
    <sequence>
     <element name="streetName" nillable="true" type="xsd:string"/>
     <element name="city" nillable="true" type="xsd:string"/>
     <element name="county" nillable="true" type="xsd:string"/>
     <element name="state" nillable="true" type="xsd:string"/>
     <element name="country" nillable="true" type="xsd:string"/>
     <element name="postalCode" nillable="true" type="xsd:string"/>
     <element name="streetNumber" nillable="true" type="xsd:string"/>
     <element name="apartmentNumber" nillable="true" type="xsd:string"/>
     <element name="neighborhood" nillable="true" type="xsd:string"/>
     <element name="reference" nillable="true" type="xsd:string"/>
     <element name="telephoneNumber" nillable="true" type="xsd:string"/>
     <element name="email" nillable="true" type="xsd:string"/>
     <element name="sequenceNumber" type="xsd:long"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseStore">
    <complexContent>
     <extension base="tns5:BaseUser">
      <sequence>
       <element name="storeId" type="xsd:int"/>
       <element name="address" nillable="true" type="tns5:Address"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseCompany">
    <complexContent>
     <extension base="tns5:BaseUser">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="address" nillable="true" type="tns5:Address"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseLineItem">
    <sequence>
     <element name="sku" nillable="true" type="xsd:string"/>
     <element name="returned" type="xsd:boolean"/>
     <element name="department" nillable="true" type="xsd:string"/>
     <element name="description" nillable="true" type="xsd:string"/>
     <element name="qty" type="xsd:double"/>
     <element name="uniqueId" nillable="true" type="xsd:string"/>
     <element name="weighted" type="xsd:boolean"/>
    </sequence>
```

```
   </complexType>
   <complexType abstract="true" name="BaseTaxInformation">
    <sequence>
     <element name="id" nillable="true" type="xsd:string"/>
     <element name="taxDescription" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseTransaction">
    <sequence>
     <element name="registerId" type="xsd:int"/>
     <element name="trxNumber" type="xsd:long"/>
     <element name="businessDate" nillable="true" type="xsd:dateTime"/>
     <element name="cashierId" type="xsd:long"/>
    </sequence>
   </complexType>
   <complexType abstract="true" name="BaseTenderType">
    <sequence>
     <element name="description" nillable="true" type="xsd:string"/>
     <element name="id" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://mgmt.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://common.ws.einvoice.pos.sap.com"/>
   <import namespace="http://parameter.ws.einvoice.pos.sap.com"/>
   <import namespace="http://external.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <complexType name="ArrayOf_tns5_nillable_Address">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="Address" nillable="true"
type="tns5:Address"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOf_tns4_nillable_TaxInformation">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="TaxInformation"
nillable="true" type="tns4:TaxInformation"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOf_tns4_nillable_LineItem">
```

```
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="LineItem" nillable="true"
type="tns4:LineItem"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOf_tns4_nillable_TaxTypeSubTotal">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="TaxTypeSubTotal"
nillable="true" type="tns4:TaxTypeSubTotal"/>
   </sequence>
   </complexType>
   <element name="insertEInvoiceResponse">
    <complexType>
     <sequence>
      <element name="insertEInvoiceReturn" nillable="true" type="tns3:WSOutput"/>
     </sequence>
    </complexType>
   </element>

   <element name="retrieveEInvoiceByKey">
    <complexType>
     <sequence>
      <element name="eInvoiceKeyInput" nillable="true" type="tns2:EInvoiceKeyInput"/>
     </sequence>
    </complexType>
   </element>
   <element name="retrieveEInvoiceByKeyResponse">
    <complexType>
     <sequence>
      <element name="retrieveEInvoiceByKeyReturn" nillable="true"
type="tns2:EInvoiceOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="retrieveEInvoiceByTransaction">
    <complexType>
     <sequence>
      <element name="transactionCriteriaInput" nillable="true"
type="tns2:TransactionCriteriaInput"/>
     </sequence>
    </complexType>
   </element>
   <element name="retrieveEInvoiceByTransactionResponse">
    <complexType>
     <sequence>
      <element name="retrieveEInvoiceByTransactionReturn" nillable="true"
type="tns2:EInvoiceOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="retrieveTransaction">
    <complexType>
     <sequence>
      <element name="transactionCriteriaInput" nillable="true"
type="tns2:TransactionCriteriaInput"/>
     </sequence>
    </complexType>
   </element>
   <element name="retrieveTransactionResponse">
    <complexType>
     <sequence>
      <element name="retrieveTransactionReturn" nillable="true"
type="tns2:TransactionOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="cancelEInvoice">
    <complexType>
     <sequence>
      <element name="eInvoiceCancellationInput" nillable="true"
type="tns2:EInvoiceCancellationInput"/>
     </sequence>
    </complexType>
   </element>
   <element name="cancelEInvoiceResponse">
    <complexType>
     <sequence>
      <element name="cancelEInvoiceReturn" nillable="true" type="tns3:WSOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="insertEInvoice">
```

```
    <complexType>
     <sequence>
      <element name="eInvoiceInput" nillable="true" type="tns2:EInvoiceInput"/>
     </sequence>
    </complexType>
   </element>
   <complexType name="ArrayOf_tns4_nillable_TenderType">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="TenderType" nillable="true"
type="tns4:TenderType"/>
    </sequence>
   </complexType>
  </schema>
 </wsdl:types>

   <wsdl:message name="insertEInvoiceResponse">
     <wsdl:part element="impl:insertEInvoiceResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="insertEInvoiceRequest">
     <wsdl:part element="impl:insertEInvoice" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="retrieveEInvoiceByTransactionRequest">
     <wsdl:part element="impl:retrieveEInvoiceByTransaction" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="retrieveTransactionResponse">
     <wsdl:part element="impl:retrieveTransactionResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="cancelEInvoiceRequest">
     <wsdl:part element="impl:cancelEInvoice" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="retrieveTransactionRequest">
```

```
     <wsdl:part element="impl:retrieveTransaction" name="parameters"/>

  </wsdl:message>


   <wsdl:message name="retrieveEInvoiceByKeyRequest">
     <wsdl:part element="impl:retrieveEInvoiceByKey" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="retrieveEInvoiceByKeyResponse">
     <wsdl:part element="impl:retrieveEInvoiceByKeyResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="cancelEInvoiceResponse">
     <wsdl:part element="impl:cancelEInvoiceResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="retrieveEInvoiceByTransactionResponse">
     <wsdl:part element="impl:retrieveEInvoiceByTransactionResponse"
name="parameters"/>

  </wsdl:message>

   <wsdl:portType name="EInvoiceMgmt">
     <wsdl:operation name="insertEInvoice">
       <wsdl:input message="impl:insertEInvoiceRequest" name="insertEInvoiceRequest"/>

       <wsdl:output message="impl:insertEInvoiceResponse"
name="insertEInvoiceResponse"/>

    </wsdl:operation>

     <wsdl:operation name="retrieveEInvoiceByKey">
       <wsdl:input message="impl:retrieveEInvoiceByKeyRequest"
name="retrieveEInvoiceByKeyRequest"/>

       <wsdl:output message="impl:retrieveEInvoiceByKeyResponse"
name="retrieveEInvoiceByKeyResponse"/>
```

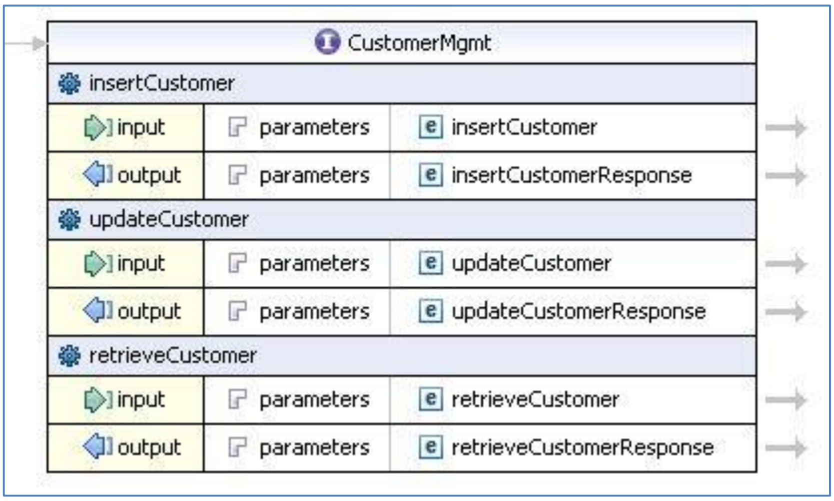
```
    </wsdl:operation>

     <wsdl:operation name="retrieveEInvoiceByTransaction">
       <wsdl:input message="impl:retrieveEInvoiceByTransactionRequest"
name="retrieveEInvoiceByTransactionRequest"/>

       <wsdl:output message="impl:retrieveEInvoiceByTransactionResponse"
name="retrieveEInvoiceByTransactionResponse"/>

    </wsdl:operation>

     <wsdl:operation name="retrieveTransaction">
       <wsdl:input message="impl:retrieveTransactionRequest"
name="retrieveTransactionRequest"/>

       <wsdl:output message="impl:retrieveTransactionResponse"
name="retrieveTransactionResponse"/>

    </wsdl:operation>

     <wsdl:operation name="cancelEInvoice">
       <wsdl:input message="impl:cancelEInvoiceRequest" name="cancelEInvoiceRequest"/>

       <wsdl:output message="impl:cancelEInvoiceResponse"
name="cancelEInvoiceResponse"/>

    </wsdl:operation>

  </wsdl:portType>

   <wsdl:binding name="EInvoiceMgmtSoapBinding" type="impl:EInvoiceMgmt">
     <wsdlsoap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>

     <wsdl:operation name="insertEInvoice">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="insertEInvoiceRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>
       <wsdl:output name="insertEInvoiceResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

     <wsdl:operation name="retrieveEInvoiceByKey">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="retrieveEInvoiceByKeyRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="retrieveEInvoiceByKeyResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

     <wsdl:operation name="retrieveEInvoiceByTransaction">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="retrieveEInvoiceByTransactionRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

      <wsdl:output name="retrieveEInvoiceByTransactionResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

     <wsdl:operation name="retrieveTransaction">
       <wsdlsoap:operation soapAction=""/>
```

```
       <wsdl:input name="retrieveTransactionRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="retrieveTransactionResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

     <wsdl:operation name="cancelEInvoice">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="cancelEInvoiceRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="cancelEInvoiceResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

  </wsdl:binding>

   <wsdl:service name="EInvoiceMgmtService">
     <wsdl:port binding="impl:EInvoiceMgmtSoapBinding" name="EInvoiceMgmt">
       <wsdlsoap:address
location="http://localhost:9080/EInvoiceMgmtWebService/services/EInvoiceMgmt"/>

    </wsdl:port>

  </wsdl:service>

</wsdl:definitions>
```

7.3 EInvoice Customer Mgmt Interface

SAP EInvoice implemented this client-side web service interface to allow third-party applications to insert, retrieve, and update customers into and from the SAP EInvoice database. SAP implemented the server-side and the retailer/vendor must implement the client-side.

The following diagram is a high-level view of the EInvoice Customer Mgmt web service interface. The third party vendor should abide by the rules defined by the interface.

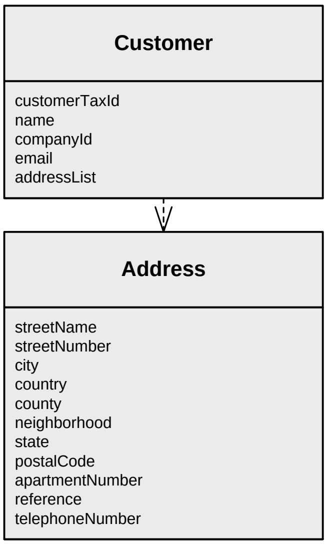


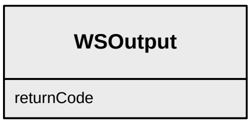
CustomerMgmt		
insertCustomer		
input	parameters	insertCustomer
output	parameters	insertCustomerResponse
updateCustomer		
input	parameters	updateCustomer
output	parameters	updateCustomerResponse
retrieveCustomer		
input	parameters	retrieveCustomer
output	parameters	retrieveCustomerResponse

7.3.1 insertCustomer

The retailer/vendor's application calls this method to create a customer in SAP's central customer database.

This method has the following signature:

Input	

Output	

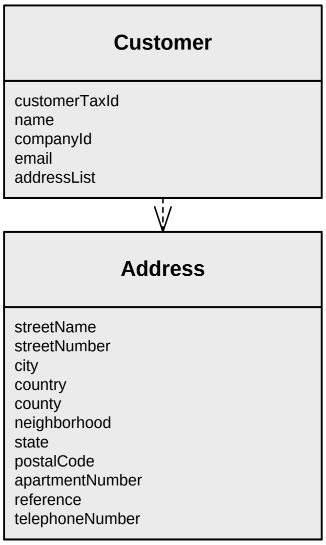
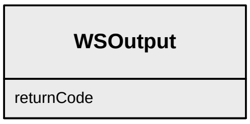
The return codes that are used for this method are:

1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

7.3.2 updateCustomer

The retailer/vendor's application shall call this method in order to update a customer record in SAP's central customer database.

This method has the following signature:

Input	 Customer customerTaxId name companyId email addressList Address streetName streetNumber city country county neighborhood state postalCode apartmentNumber reference telephoneNumber
Output	

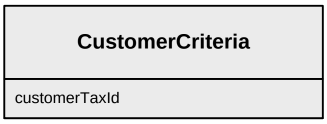
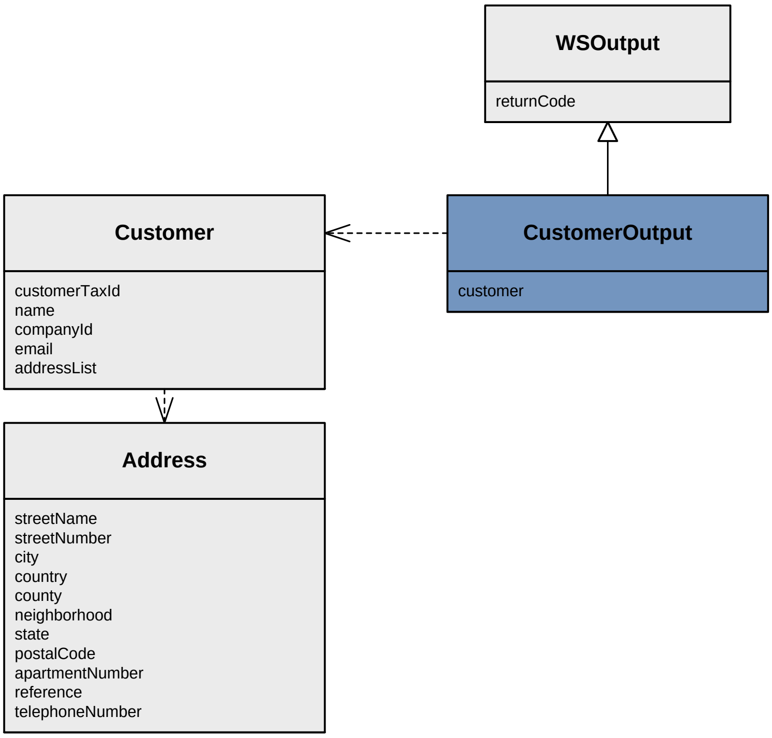
The return codes that are used for this method are:

1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

7.3.3 retrieveCustomer

The retailer/vendor's application calls this method to retrieve a customer record in SAP's central customer database. A customer can be retrieved by providing any of the customer attributes.

This method has the following signature:

Input	
Output	 WSOutput returnCode Customer customerTaxId name companyId email addressList CustomerOutput customer Address streetName streetNumber city country county neighborhood state postalCode apartmentNumber reference telephoneNumber

The return codes that are used for this method are:

1. `Success`
2. `EInvoice does not exist`
3. `Internal error`

Below is a WSDL that defines the EInvoice Customer Mgmt web service interface.

```
<?xml version="1.0" encoding="UTF-8"?>

<wsdl:definitions targetNamespace="http://mgmt.ws.customer.pos.sap.com"
xmlns:impl="http://mgmt.ws.customer.pos.sap.com"
xmlns:intf="http://mgmt.ws.customer.pos.sap.com"
xmlns:tns2="http://parameter.ws.einvoice.pos.sap.com"
xmlns:tns3="http://common.ws.einvoice.pos.sap.com"
xmlns:tns4="http://external.model.einvoice.pos.sap.com"
xmlns:tns5="http://common.model.einvoice.pos.sap.com"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
```

```
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsi="http://ws-
i.org/profiles/basic/1.1/xsd" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <wsdl:types>
  <schema targetNamespace="http://parameter.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://external.model.einvoice.pos.sap.com"/>
   <import namespace="http://common.ws.einvoice.pos.sap.com"/>
   <complexType name="CustomerInput">
    <complexContent>
     <extension base="tns3:WSInput">
      <sequence>
       <element name="customer" nillable="true" type="tns4:Customer"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="CustomerCriteriaInput">
    <complexContent>
     <extension base="tns3:WSInput">
      <sequence>
       <element name="customerCriteria" nillable="true" type="tns2:CustomerCriteria"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType name="CustomerCriteria">
    <sequence>
     <element name="taxId" nillable="true" type="xsd:string"/>
     <element name="companyId" type="xsd:int"/>
    </sequence>
   </complexType>
   <complexType name="CustomerOutput">
    <complexContent>
     <extension base="tns3:WSOutput">
      <sequence>
       <element name="customer" nillable="true" type="tns4:Customer"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.ws.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="WSInput">
    <sequence>
     <element name="sourceSystemCode" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="WSOutput">
    <sequence>
     <element name="returnCode" type="xsd:int"/>
     <element name="returnMessage" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://external.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://mgmt.ws.customer.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <complexType name="Customer">
    <complexContent>
     <extension base="tns5:BaseCustomer">
      <sequence>
       <element name="addressList" nillable="true"
type="impl:ArrayOf_tns5_nillable_Address"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
  </schema>
  <schema targetNamespace="http://common.model.einvoice.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType abstract="true" name="BaseCustomer">
    <complexContent>
     <extension base="tns5:BaseUser">
      <sequence>
       <element name="companyId" type="xsd:int"/>
       <element name="insertDate" nillable="true" type="xsd:dateTime"/>
       <element name="lastChangedDate" nillable="true" type="xsd:dateTime"/>
       <element name="insertBy" nillable="true" type="xsd:string"/>
       <element name="lastChangedBy" nillable="true" type="xsd:string"/>
       <element name="customerType" nillable="true" type="xsd:string"/>
      </sequence>
     </extension>
    </complexContent>
   </complexType>
   <complexType abstract="true" name="BaseUser">
    <sequence>
     <element name="taxId" nillable="true" type="xsd:string"/>
     <element name="name1" nillable="true" type="xsd:string"/>
     <element name="name2" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="Address">
    <sequence>
     <element name="streetName" nillable="true" type="xsd:string"/>
     <element name="city" nillable="true" type="xsd:string"/>
     <element name="county" nillable="true" type="xsd:string"/>
     <element name="state" nillable="true" type="xsd:string"/>
     <element name="country" nillable="true" type="xsd:string"/>
     <element name="postalCode" nillable="true" type="xsd:string"/>
     <element name="streetNumber" nillable="true" type="xsd:string"/>
     <element name="apartmentNumber" nillable="true" type="xsd:string"/>
     <element name="neighborhood" nillable="true" type="xsd:string"/>
     <element name="reference" nillable="true" type="xsd:string"/>
     <element name="telephoneNumber" nillable="true" type="xsd:string"/>
     <element name="email" nillable="true" type="xsd:string"/>
     <element name="sequenceNumber" type="xsd:long"/>
    </sequence>
   </complexType>
  </schema>
  <schema targetNamespace="http://mgmt.ws.customer.pos.sap.com"
xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://common.ws.einvoice.pos.sap.com"/>
   <import namespace="http://parameter.ws.einvoice.pos.sap.com"/>
   <import namespace="http://common.model.einvoice.pos.sap.com"/>
   <complexType name="ArrayOf_tns5_nillable_Address">
    <sequence>
     <element maxOccurs="unbounded" minOccurs="0" name="Address" nillable="true"
type="tns5:Address"/>
    </sequence>
   </complexType>
   <element name="insertCustomerResponse">
    <complexType>
     <sequence>
      <element name="insertCustomerReturn" nillable="true" type="tns3:WSOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="updateCustomer">
    <complexType>
     <sequence>
      <element name="customerInput" nillable="true" type="tns2:CustomerInput"/>
     </sequence>
    </complexType>
   </element>
   <element name="updateCustomerResponse">
    <complexType>
     <sequence>
      <element name="updateCustomerReturn" nillable="true" type="tns3:WSOutput"/>
     </sequence>
    </complexType>
   </element>
<element name="retrieveCustomer">
    <complexType>
     <sequence>
      <element name="customerCriteriaInput" nillable="true"
type="tns2:CustomerCriteriaInput"/>


     </sequence>
    </complexType>
   </element>
   <element name="retrieveCustomerResponse">
    <complexType>
     <sequence>
      <element name="retrieveCustomerReturn" nillable="true"
type="tns2:CustomerOutput"/>
     </sequence>
    </complexType>
   </element>
   <element name="insertCustomer">
    <complexType>
     <sequence>
      <element name="customerInput" nillable="true" type="tns2:CustomerInput"/>
     </sequence>
    </complexType>
   </element>
  </schema>
 </wsdl:types>

   <wsdl:message name="retrieveCustomerRequest">
     <wsdl:part element="impl:retrieveCustomer" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="retrieveCustomerResponse">
     <wsdl:part element="impl:retrieveCustomerResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="updateCustomerResponse">
     <wsdl:part element="impl:updateCustomerResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="updateCustomerRequest">
     <wsdl:part element="impl:updateCustomer" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="insertCustomerResponse">
     <wsdl:part element="impl:insertCustomerResponse" name="parameters"/>

  </wsdl:message>

   <wsdl:message name="insertCustomerRequest">
     <wsdl:part element="impl:insertCustomer" name="parameters"/>

  </wsdl:message>

   <wsdl:portType name="CustomerMgmt">
     <wsdl:operation name="insertCustomer">
       <wsdl:input message="impl:insertCustomerRequest" name="insertCustomerRequest"/>
```

```
       <wsdl:output message="impl:insertCustomerResponse"
name="insertCustomerResponse"/>

    </wsdl:operation>

     <wsdl:operation name="updateCustomer">
       <wsdl:input message="impl:updateCustomerRequest" name="updateCustomerRequest"/>

       <wsdl:output message="impl:updateCustomerResponse"
name="updateCustomerResponse"/>

    </wsdl:operation>

     <wsdl:operation name="retrieveCustomer">
       <wsdl:input message="impl:retrieveCustomerRequest"
name="retrieveCustomerRequest"/>

       <wsdl:output message="impl:retrieveCustomerResponse"
name="retrieveCustomerResponse"/>

    </wsdl:operation>

  </wsdl:portType>

   <wsdl:binding name="CustomerMgmtSoapBinding" type="impl:CustomerMgmt">
     <wsdlsoap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>

     <wsdl:operation name="insertCustomer">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="insertCustomerRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="insertCustomerResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>
    </wsdl:operation>

     <wsdl:operation name="updateCustomer">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="updateCustomerRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="updateCustomerResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

     <wsdl:operation name="retrieveCustomer">
       <wsdlsoap:operation soapAction=""/>

       <wsdl:input name="retrieveCustomerRequest">
         <wsdlsoap:body use="literal"/>

      </wsdl:input>

       <wsdl:output name="retrieveCustomerResponse">
         <wsdlsoap:body use="literal"/>

      </wsdl:output>

    </wsdl:operation>

  </wsdl:binding>

   <wsdl:service name="CustomerMgmtService">
     <wsdl:port binding="impl:CustomerMgmtSoapBinding" name="CustomerMgmt">
       <wsdlsoap:address
location="http://localhost:9080/CustomerMgmtWebService/services/CustomerMgmt"/>

    </wsdl:port>
  </wsdl:service>

</wsdl:definitions>
```

8 User Guide

8.1 Logon and Site Authorization

Cashiers and managers should have the required authorization to access the POS Client.

8.2 User Roles in EInvoice

The manager and cashier user roles are supported for EInvoicing transactions. Throughout this document we use the term “user” or “you” to refer to a manager or cashier. We use a specific role designation if a task or activity is restricted by role.

8.3 EInvoice Functionality

EInvoice functionality in SAP POS is based on the following assumptions:

- During a sales transaction, a customer must be asked if an EInvoice is required.
- Only one EInvoice can be generated for a sales transaction.
- The EInvoice must be generated and issued on the same day that the sales transaction was processed in SAP POS.
- An EInvoice can be printed, reprinted, or canceled only for sales transactions generated in SAP POS after the EInvoice application is deployed. The migration or upgrade of historic transactions is not possible as additional data is required that cannot be generated.
- A sales transaction can be either fully (all items) or partially (selected items) invoiced.
- An EInvoiced item cannot be returned at a later date.
- Any transaction that includes an EInvoiced item cannot be post-voided.

The EInvoice 2.7 software covers the following activities conducted by a user from a POS terminal at a store:

- Print an EInvoice
- Reprint an EInvoice
- Cancel an EInvoice.

8.4 Screen Functionality Overview

The following buttons are available on Web pages for the EInvoice application:

Button	Where	Action
Electronic Invoice	Main Menu (POS Client)	Access POS Client function submenu.
Done (F1)	Submenu (POS Client)	For printing and reprinting an EInvoice: • If selected while on the confirmation page: triggers printing or reprinting of the EInvoice. • If selected while on any page other than the confirmation page: cancels the print or reprint and closes the browser window. For canceling an EInvoice: stops the cancellation and closes the browser window.
Quick eInvoice (F1) *Print* (F2) *Reprint* (F3) *Cancel* (F4)	Submenu (POS Client)	Launch EInvoice Web application for one of the following functions: • Generate and Print • Reprint • Cancel.
Cancel Invoice (F7)	Web page for EInvoice	Cancel the EInvoice.
Generate Invoice (F7)	Web page for EInvoice	Confirm the generation of an EInvoice for the selected items in the sales transaction and for the specified customer.
New Customer (F10)	Web page for EInvoice	Clears all customer information in the customer information data fields (Customer Information Entry page).
New Address (F2)	Web page for EInvoice	Clears all customer address information in the customer information data fields (Customer Information Entry page).
Next (F9)	Web page for EInvoice	Continue the transaction.
Previous (F8)	Web page for EInvoice	Stop the transaction and return to the previous page.
Search (F3) *Search by Tax Id* (F3) *Search by Name* (F4)	Web page for EInvoice	Search for a customer (*Customer Information Entry* page). Search for transaction (*Search Transaction* page and *Search Result* page).

8.5 EInvoice Navigation with a Keyboard

Navigation works the same for all Web pages of the EInvoice application. However, if a cash register is not equipped with a touch screen or a touch pad, you use the keyboard to navigate in the Web pages.

Keyboard Key	Result
ENTER	Move ahead to the next field.
UP ARROW	Move back to the previous field.
DOWN ARROW	Move ahead to the next field.
Function keys (F1, F2, F3, and so on)	Execute the action described by the function key description. For example, launch a search with *Search* (F3).

8.6 EInvoice Screens

8.6.1 SAP POS Screens

You launch the EInvoice application from the Main menu.

1. From the Main menu, choose Electronic Invoice.

   The Submenu displays the Quick eInvoice, Print, Reprint and Cancel options.

2. Choose the relevant option:
   - *Quick eInvoice* (F1)
   - *Print* (F2)
   - *Reprint* (F3)
   - *Cancel* (F4)

   The Web browser opens and the search page for the selected function of the EInvoice application appears.

8.6.2 EInvoice Pages

The following pages are displayed:

Page	Activity
Search Transaction	Search for a sales transaction for which to generate, print, reprint, or cancel an EInvoice. This page is not displayed when the *Quick eInvoice* option is selected.
Search Result	Display a sales transaction that meets the search criteria specified in the *Search Transaction* page.
*Invoice Creation – Item Selection**	Display sales transaction item(s). Select items to include in EInvoice.
Customer Information Entry	• Search for existing customer. • Display results of customer search.

Page	Activity
	- Create a new customer and specify customer data. - Generate and print an EInvoice.
Confirmation Page	Show the selected element before performing the required action.
Error Page	Displays a message that the operation could not be completed.
Session Timeout	Displays a message that the EInvoice session has expired.

* The default version of the application does not display this function. It is available only upon specific configuration.

8.7 Using Quick EInvoice

The *Quick eInvoice* option provides a quick way to generate an electronic invoice. When you choose this option, the POS uses details from the last logged sales transaction and displays them on the screen. This lets you generate and then print an EInvoice without having to search for the correct transaction first.

Prerequisites

- A sales transaction has been carried out in SAP POS 2.3 but an EInvoice has not yet been generated. The customer requests that you print an EInvoice immediately following their sales transaction.

Procedure

To generate and print a Quick EInvoice for a customer, you use the details from the previous transaction that are displayed on the screen, and then add the customer name and address information to it.

1. From the Main menu, choose *Electronic Invoice* (`F8`), and from the submenu choose *Quick Invoice* (`F1`).
2. The customer search page appears where you can see the items from the previous sales transaction.
3. If this is the transaction that you need to generate the electronic invoice for, enter the customer information for this sale. You can do one of the following:
   - Search for a customer by entering their name in the *Enter Tax Id or Customer Name* field and then choosing *Search by Name* (`F4`).
   - Search for a customer by entering their tax ID number in the *Enter Tax Id or Customer Name* field and then choosing *Search by Tax ID* (`F3`).
   - Create a new customer record by choosing *New Customer* (`F10`) and then entering the tax ID name and address information for the new customer.
4. Verify the transaction information. If this is not the correct transaction, choose *Previous* (`F8`) to return to the transaction search page.
5. The remainder of the process is as printing a regular EInvoice. For more information about the printing process, see section *8.8*, *Generating and Printing an EInvoice from a POS Terminal*.

8.8 Generating and Printing an EInvoice from a POS Terminal

This functionality lets a cashier or manager generate and then print an EInvoice at a POS terminal with SAP POS installed.

Prerequisites

- A sales transaction has been carried out in SAP POS 2.3 but an EInvoice has not yet been generated. The customer requests that you print an EInvoice on the day of the sales transaction and presents the sales receipt.
- In this version, default settings have been made that will not show the transaction item search page, however you can choose to display it. If you choose so, you will need to change the required settings in the configuration file.

Procedure

To generate and print an EInvoice for a customer, you must first find their sales transaction and then add their name and address information to the invoice.

1. From the Main menu, choose *Electronic Invoice* (F8), and from the submenu choose *Print* (F2). The page for identifying the sales transaction appears.
2. To find a sales transaction, enter the required information, press Enter and then choose *Search* (F3).

   Many of the search criteria such as the store number, register number, and business date are prefilled on the screen to help you locate the transaction faster. If any of these parameters is missing, or if you need to look for a different transaction, press Enter. This brings up the next parameter in the list. Each time you press Enter, the field description changes to the next one in the search criteria list. Alternately, you can also use the Up and Down arrow keys.

   Note

   - If you made settings to display the transaction search results and the item selection page, you will see them next. See *8.8.1, Use Case: Generating an EInvoice with Item Selection* for more information on this feature.
   - Otherwise, the customer search page appears. Verify the transaction information. If this is not the correct transaction, click *Previous* (F8) to return to the transaction search page.

   You can search for a customer record in any of the following ways:

   - By entering a tax ID number
   - By entering a customer name
3. Depending on the search method you select, you must do one of the following:

- To search for a customer using a tax ID, enter the tax ID number in the *Tax Id* field then choose *Search by Tax ID* (F3).

  If the tax ID is valid, the page showing the customer information associated with the tax ID appears.

  Note

  Generic tax IDs

  You cannot execute a customer search when a customer is providing a generic tax ID unless you enter the customer information first.

  When a regular tax ID or foreign customer generic tax ID is used, the EInvoice displays the net price for an item. If a generic resident tax ID is used, instead of the net price the gross price is displayed. The gross price includes the tax. Additionally, for generic resident tax IDs each tax line shows 0.00 instead of the IVA taxes which are broken down and displayed on invoices for the regular or generic foreign tax IDs.

  Note

  The data you enter in the *RFC* and the *Zip Code* fields must conform to the following requirements:

  - *RFC* format is `CCCCNNNNNNAAA` or `CCCNNNNNNAAA` where:

        - `CCCC` is for individuals and `CCC` is for corporations and must be alpha characters A–Z and the special character &.
        - `NNNNNN` must be in the date format `YYMMDD`.
        - `AAA` must be alphanumeric characters (letters and numbers) or the special character _ (underscore).
    - *Zip Code*: Must be 5 digits. Validation of the *Zip Code* field is configurable; see section *5.1, EInvoice Configuration Properties File*.
- To search for a customer using their name, enter the name in the *Customer Name* field, and then choose *Search by Name* (F4). You do not need to enter the complete name to launch the search; you can enter only the first few letters of the name or the first initial.

  The system displays the records closest to the letters you entered ordered by tax ID.

  Depending on the status of the customer record one of the following scenarios occur, which determine what you can do next.

4. Continue with one of the following steps:

Customer information status	What you can do
The customer information is found.	Verify the customer address information. If it is correct, you are ready to generate the invoice. If the customer has more than one address, choose *Next Address* (F12) to advance to the next address record in the database, or *Previous Address* (F11) to return to the previous one.
The customer information is found but the customer has a new address.	Choose *New Address* (F2) to enter the new address information. When you have entered all the required information, you are ready to generate the invoice. Note The customer information page does not have a save function. Any customer data you enter is saved automatically when you generate the invoice.
There are more than one customer records with similar names or tax ID.	Choose *Next Customer* (F4) to advance to the next customer record in the database, or *Previous Customer* (F3) to return to the previous one. When you find the customer's information, you are ready to generate the invoice.
The customer information is not found.	Choose *New Customer* (F10) to create a new customer record. When you have entered all the required information, you are ready to generate the invoice. Note The customer information page does not have a save function. Any customer data

Customer information status	What you can do
	you enter is saved automatically when you generate the invoice.
More than 10 results records are found	The system will not show more than 10 records for the search criteria entered. If your selection produces more than 10 records, refine your search criteria.

5. Choose *Generate Invoice* (F7).

   When you choose *Generate Invoice*, the new address or the new customer information is added to the database. A confirmation message appears with the customer information and the invoice details.

   This page shows a message that the invoice was successfully generated.

6. To print the EInvoice, click *Done* (F1 in the submenu).

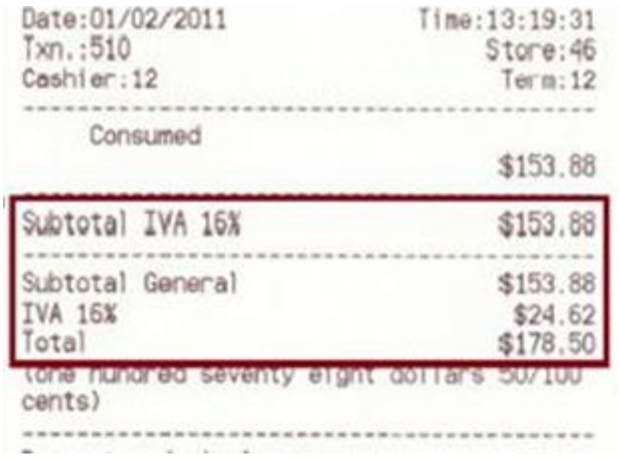
   The EInvoice is printed with all relevant information.

7. At the successful print confirmation prompt, do one of the following:

- Choose *Yes* (F1) on the POS menu to confirm that the invoice was printed successfully.
- If the invoice did not print due to a paper jam or other issues with the printer, choose *No* (F2) and try to print the invoice again after the issue is resolved.

Example

An electronic invoice generated for a regular or generic foreign tax ID:

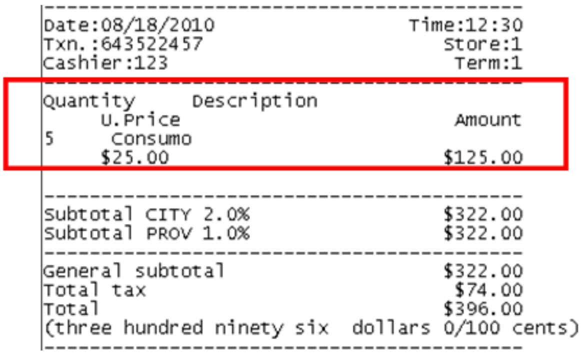


```
Date:01/02/2011                          Time:13:19:31
Txn.:510                                      Store:46
Cashier:12                                     Term:12
------------------------------------------------------
     Consumed
                                               $153.88
------------------------------------------------------
Subtotal IVA 16%                               $153.88
------------------------------------------------------
Subtotal General                               $153.88
IVA 16%                                         $24.62
Total                                          $178.50
(one hundred seventy eight dollars 50/100
cents)
------------------------------------------------------
```

Note

If items from the food and beverage department are included in the EInvoice, they are grouped in one row, the individual item amounts are added up, and the row description displays *Consumo* and not the individual item descriptions.

Example



```
------------------------------------------
Date:08/18/2010                 Time:12:30
Txn.:643522457                     Store:1
Cashier:123                         Term:1
------------------------------------------
Quantity     Description
     U.Price                        Amount
5      Consumo
     $25.00                        $125.00

------------------------------------------
Subtotal CITY 2.0%                 $322.00
Subtotal PROV 1.0%                 $322.00
------------------------------------------
General subtotal                   $322.00
Total tax                           $74.00
Total                              $396.00
(three hundred ninety six  dollars 0/100 cents)
------------------------------------------
```

8.8.1 Use Case: Generating an EInvoice with Item Selection

In the default version of the application, settings have been made that will not show the transaction *Search Results* page and the *Item Selection* page, however you can choose to display them. To do so you must change the required settings in the configuration file.

The following steps describe how to work with the *Transaction Search Results* page and the *Item Selection* list.

1. In the *Transaction Search Results* page, confirm that the transaction found is the correct one and then choose *New Invoice* (F7).
2. In the *Transaction Items* list, select the eligible items to be invoiced.
   - To select all eligible items, choose *Select All* (F4).
   - To select a subset of eligible items, in the *Selection* column for an item click the checkbox.
     - Choose *Select* (F3) after making your selections.

       The items you selected (either all items or a subset of items) are moved from the *Transaction Items* list to the *Selected Items* to be invoiced list. The *Net Price* and *Net Total* are displayed for each item selected. The subtotal of the selected items plus relevant taxes are displayed, along with a total amount for the selected items.
3. Verify the information. If the selected items to be invoiced are not correct, you can remove some or all of the items from the list.
   - To remove all items, choose *Remove All* (F12). Select other items or return to the search page by choosing *Previous* (F8).
   - To select one or more items to be removed, in the *Selection* column for an item click the checkbox. To remove the selected items, choose *Remove* (F11).
4. To continue, choose *Next* (F9). To return to the *Transaction Selection* page, choose *Previous* (F8).
5. The customer information page appears.
6. The remainder of the process is as printing a regular EInvoice. For more information about the printing process, see section *8.8, Generating and Printing an EInvoice from a POS Terminal*.

8.9 Reprinting an EInvoice from a POS Terminal

This function lets a cashier or manager reprint an EInvoice at a POS terminal with SAP POS 2.3 installed. There is no limit to the number of times that you can reprint an EInvoice.

Prerequisites

A sales transaction has been carried out in SAP POS. An EInvoice for the transaction has been printed at least once, and the EInvoice has not been canceled.

To search for an invoice to reprint, you need either the folio information or the transaction details. The customer must provide a sales receipt or you can search for an invoiced transaction. The results page displays the information you need.

Procedure

1. From the Main menu, choose *Electronic Invoice* (F8), and from the submenu choose *Reprint* (F3).

   The search page for reprinting an electronic invoice displays.

2. To find an EInvoice for a sales transaction, you can search by the following folio information (F4) or transaction information:

   - Folio prefix and folio number assigned to the EInvoice
   - Transaction details, including store, register, business date and transaction number

   Folio fields have the higher priority. If you enter both folio information and transaction details, the transaction details are ignored.

3. Specify the required information and then choose *Search* (F3).

   A confirmation page appears with customer information and invoice details showing a list of items previously invoiced.

4. Verify the information displayed.

   - If this is not the correct transaction, to cancel the reprint and return to the search page, choose *Previous* (F8).
   - If this is the correct transaction, to reprint the electronic invoice, choose *Done* (F1) in the submenu on the POS.

The electronic invoice is printed with all relevant information.

8.10 Canceling an EInvoice from a POS Terminal

This functionality lets a cashier or manager cancel an EInvoice from a POS terminal with SAP POS installed. A customer can request that you cancel an EInvoice for the following reasons:

- To return an item that was previously invoiced.
- To correct the item selection on an EInvoice.

Prerequisites

A sales transaction has been carried out in SAP POS. An EInvoice for the transaction has been printed at least once, and the EInvoice has not been canceled.

To search for an invoice to cancel, you need either the folio information or the transaction information. The customer must provide a sales receipt, or you can search for an invoiced sales transaction to find the information you need.

Procedure

At any time during the cancelation transaction, to stop the process, close the browser window and return to the Main menu, choose *Done* (F1 in the submenu).

1. From the Main menu, choose *Electronic Invoice* (F8) and from the Submenu, choose *Cancel* (F4).

   The search page for canceling an electronic invoice displays.
2. To find an EInvoice for a sales transaction, you can search by the following folio information (F4) or transaction information:
   - Folio prefix and folio number
   - Transaction details, including store, register, business date and transaction number

   Folio fields have the higher priority. If you enter the folio information and transaction details, the transaction details are ignored.
3. Specify the required information and then choose *Search* (F3).

   A page displays with customer information and invoice details showing a list of items invoiced.
4. Verify the information.
   - If this is not the correct transaction, choose *Previous* (F8) to return to the search page.
   - If this is the correct transaction, to cancel the invoice, choose *Cancel Invoice* (F7).

If the cancellation is successful, a confirmation message displays.

8.11 System Messages and Screens

8.11.1 Error Page

If a problem occurs at any time while you are generating, printing, reprinting, or canceling an electronic invoice, the *Error Page* appears. The first line on the page indicates the operation you were performing. To close the window, choose *Done* (F1 in the submenu).

8.11.2 Session Timeout Page

If an EInvoice session has no user interaction, the session times out and the *Session Timeout* page appears. This can occur whether you are generating, reprinting, or canceling an invoice. To close the window, choose *Done* (F1 in the submenu).

9 Operations Information

This section contains general information on monitoring, administration, and troubleshooting the EInvoice application. The operations information specific to the EInvoice application is described in the following sections.

9.1 Monitoring

9.1.1 Logs

EInvoice logs allow you to monitor and track errors in the EInvoice system.

Note

Websphere logs such as `SystemOut.log`, `SystemErr.log`, `startServer.log`, and `trace.log` are located in the `%WAS_HOME%/profiles\EXPRESSRETURNS\logs\server1` folder and are useful in monitoring and debugging the EInvoice application.

EInvoice log properties and locations are configured in **`<SAP POS Returns Authorization installation folder>`**`\EInvoice\EInvoiceConfig\einvoiceConfig.properties`. For more information, see section *5.1, EInvoice Configuration Properties File*.

To fully enable the EInvoice debug level logging, complete the following in Websphere:

1. Open the Websphere console.
2. Expand the *Troubleshooting* node.
3. Select *Logs and trace*.
4. In the right window pane, select *server1*.
5. Select *Change Log Detail Levels*.
6. Select the *Runtime* tab.
7. Under the *General Properties* section, enable the *Save runtime changes to configuration as well* checkbox.
8. Expand the *All Components* node.
9. Find and select the `com.sap.pos.*` node. A context menu displays.
10. Select the *Message and Trace Levels* menu. A sub-context menu displays.
11. Select *fine*.
12. Select *Apply*.
13. In the Message section at the top of the web page, select *save*.

Note

The steps below are only applicable in IBM Websphere 7. Refer to IBM's Websphere 7 or related User Guide.

9.2 Administration and Management

9.2.1 Starting and Stopping EInvoiceEAR and EInvoiceWSEAR

Configuration changes in the EInvoice configuration (such as eInvoiceConfig.properties, `eInvoiceFormatConfig.properties`, `eInvoiceReceiptDocument.form`, or `eInvoiceCheckingRuleConfig.xml`) will only take effect after the `EInvoiceEAR` and `EInvoiceWSEAR` applications are restarted.

To restart the `EInvoiceEAR` and `EInvoiceWSEAR` applications:

1. Open the *Websphere* console.
2. Expand the *Applications* node.
3. Expand the *Application Types* node.
4. Select *WebSphere enterprise* applications.
5. In the right window pane, check the *EInvoiceEAR* and *EInvoiceWSEAR* application checkboxes and choose *Stop* button.
6. Wait until *EInvoiceEAR* and *EInvoiceWSEAR* applications are stopped.
7. In the right window pane, check the *EInvoiceEAR* and *EInvoiceWSEAR* application checkboxes and choose *Start* button.
8. Wait until *EInvoiceEAR* and *EInvoiceWSEAR* applications are started.

Note

The steps below are only applicable in Websphere 7. Refer to IBM's Websphere 7 or related User Guide.

9.3 Troubleshooting

EInvoice logs allow you to troubleshoot the EInvoice application when configured at the `DEBUG`, `FINE` or `ALL` logging level. You can also view and verify the available variable names and values that can be used in the EInvoice receipt document configuration.

Key Phrase

The following are key phrases to search for in the debug level logs when troubleshooting EInvoice application. `INFO` level logs are also available.

Key Phrase	Description
`EInvoiceServiceBean} - retrieveTransactionFromRA: Request of retrieving transaction:`	This phrase is shown for every EInvoice request to retrieve a transaction from SAP Returns Authorization. This phrase is shown in the `INFO` level logs and above.

Key Phrase	Description
`EInvoiceServiceBean} - retrieveTransactionFromRA: Response of retrieving transaction:`	This phrase is shown right after the EInvoice request to retrieve a transaction from SAP Returns Authorization. This phrase is shown in the `INFO` level logs and above.
`EInvoiceServiceBean} - retrieveTransactionFromRA: Internal Transaction object after converting: Transaction`	This phrase is shown to indicate the conversion of the SAP Returns Authorization response into an EInvoice internal representation of the transaction. This phrase is shown in the `INFO` level logs and above.
`EInvoiceServiceBean} - generateEInvoiceFromWebService: Generating eInvoice by calling web service provider`	This phrase is shown to indicate that the EInvoice is about to request the Third Party Web Service Façade Server. This phrase is shown in the `INFO` level logs and above.
`EInvoiceServiceBean} - generateEInvoice: Inserting generatedEInvoice=`	This phrase is shown to indicate that the successfully generated EInvoice will be inserted into the database. This phrase is shown in the `FINE` level logs and above.
`formatEInvoice: Receipt EInvoice Document Configuration=`	This phrase is only shown once during the first EInvoice transaction right after the EInvoice start-up. This phrase is shown in the `INFO` level logs and above. The parsed and loaded EInvoice receipt document configuration file is shown right after this phrase.
`formatEInvoice: Generic Variables=`	This phrase is only shown once during the first EInvoice transaction right after EInvoice start-up. This phrase is shown in the `DEBUG` level logs and above. The list of generic variables that are used for every EInvoice transaction are shown right after this phrase.
`formatEInvoice: documentVariables=`	This phrase is shown on every EInvoice transaction. This phrase is shown in the `DEBUG` level logs and above. The list of EInvoice variables that are used for the current EInvoice transaction are shown right after this phrase.

10 Appendix

10.1 List of Well-Known Forms

Note

- Each of the Well-Known Forms listed below can be used by the EInvoice Receipt. Some of the Well-Known Forms have unique logic in handling the forms.
- All Well-Known Forms are optional. Not all of the Well-Known Forms are needed to be listed in the `format_order` property of the `eInvoiceFormatConfig.properties`. Well-Known Forms not listed in the `format_order` property will not be printed.
- If a Well-Known Form is listed in the format_order property, but not configured in the `eInvoiceReceiptDocument.form` file the form will be ignored.
- All EInvoice variables and generic variables are available in all forms, unless otherwise specified.

Well-Known Form	Description
`HEADER`	The receipt header section of the receipt. Header related information can be placed in this form. Note Any content in this form is aligned in the center and printed with the value specified in the `doubleHighWideChar` property of `eInvoiceFormatConfig.properties` file.
`COMPANY_INFO`	The company information section of the receipt. Company related information is placed in this form, such as, but not limited to company name and address information.
`STORE_INFO`	The store information section of the receipt. Store related information is placed in this form, such as, but not limited to store and address information.
`FISCAL_REGIME`	The fiscal regime section of the receipt. Fiscal related information is placed in this form.
`FOLIO_INFO`	The folio information section of the receipt. Folio related information is placed in this form, such as, but not limited to folio prefix and folio number.
`ISSUE_INFO`	The EInvoice issue information section of the receipt. Information, such as, but not limited to the place and date of issue of the current electronic invoice is placed in this form.

Well-Known Form	Description
CUSTOMER_INFO	The customer information section of the receipt. Customer related information is placed in this form, such as, but not limited to customer and address information.
TRX_DATA	The transaction information section of the receipt. Sales transaction related information is placed in this form; such as but not limited to store number, business date, transaction number, terminal number, cashier number, etc.
ITEM_HEADER	The item header form of the receipt. This would typically contain the labels for the columns of item information to be printed for each item in the item data (or detail) sections.
ITEM_DATA	The items data section of the receipt. The invoiced items in this form include all regular items and all food and beverage items. Item related information is placed in this form, such as item identifier, item description, etc. Note This form will be processed once for each invoiced item in the transaction.
REGULAR_ITEMS_DETAIL	The regular items section of the receipt. Item related information is placed in this form, such as item identifier, item description, etc. Note This form will be processed once for each invoiced regular item in the transaction.
FB_ITEMS_DETAIL	The food and beverages items data section of the receipt. Item related information is placed in this form, such as item identifier, item description, etc. Note This form will be processed once for each invoiced food and beverage item in the transaction.
ITEM_FOOTER	The item footer section of the receipt. Contains general regular and food and beverage item related information.
TAXABLE_SUBTOTAL_DATA	The taxable subtotal data section of the receipt. Taxable subtotal related information is placed in this form, such as tax description, and taxable subtotal amount. Note This form will be processed once for each tax ID taxable subtotal amount.

Well-Known Form	Description
`SUBTOTAL_GENERAL`	The subtotal general information section of the receipt. Subtotal general information is placed in this form, such as, but not limited to, EInvoice grand total amount or EInvoice net amount.
`TAX_TOTALS_DATA`	The tax total data section of the receipt. Tax totals data-related information is placed in this form, such as tax ID, tax description, tax rate, tax amount, and so on. Note This form will be processed once for each tax total data.
`TOTAL`	The total information section of the receipt. Total related information is placed in this form, such as, but not limited to, EInvoice grand total amount.
`TENDER_TYPE`	The tender type section of the receipt. Tender type related information is placed in this form, such as tender description, account number, tender amount, etc. Note This form will be processed for each tender type in the transaction.
`FIXED_TEXT`	The fixed text section of the receipt.
`APPROVAL_DATA`	The approval data section of the receipt. Approval data related information is placed in this form, such as, but not limited to, approval number, approval year, and certificate number.
`OIM`	The original input message section of the form. Original input message text related information is placed in this form, such as, but not limited to, original input message field.
`DIGITAL_SEAL`	The digital seal section of the receipt. Digital seal related information is placed in this form, such as, but not limited to, digital field.

10.2 List of Document Variables Derived from EInvoice Data Fields

Listed below are the available EInvoice variables that can be used in the EInvoice receipt document configuration. The value of these variables is unique for every EInvoice generation transaction.

Note

Variables are case-sensitive.

Document Variables	Description
`%trxStoreId%`	Sale transaction store ID
`%trxRegisterId%`	Sale transaction register ID
`%trxNumber%`	Sale transaction number
`%trxBusinessDate%`	Sale transaction business date
`%trxBusinessTime%`	Sale transaction business date-time
`%trxCashierId%`	Sale transaction cashier ID
`%insertBy%`	This invoice inserted by a cashier ID
`%cancelBy%`	This invoice cancelled by a cashier ID
`%comments%`	EInvoice comments
`%trxOriginCode%`	Transaction original code
`%cancelled%`	Cancelled flag
`%cancelDate%`	Date of called EInvoice
`%originalInputMessage%`	Original input message
`%digitalSeal%`	EInvoice digital seal
`%UUID%`	UUID
`%stampDate%`	Stamp Date
`%stampTime%`	Stamp Time
`%SATCertificateNumber%`	SAT Certificate Number
`%SATDigitalSeal%`	SAT Digital Seal
`%TFDOriginalChain%`	TFD Original Chain
`%approvalNumber%`	EInvoice approval number
`%approvalYear%`	EInvoice year of approval
`%certificateNumber%`	EInvoice certificate number
`%invoiceDate%`	Date of the invoice
`%invoiceDateTime%`	Date and time of the invoice

Document Variables	Description
`%amountOnText%`	EInvoice amount in text format
`%documentType%`	EInvoice document type
`%xmlPath%`	XML path returned by EInvoice third party web service server
`%pdfPath%`	PDF path returned by EInvoice third party web service server
`%eInvoiceTaxAmount%`	EInvoice tax amount
`%eInvoiceNetAmount%`	EInvoice net amount
`%eInvoiceSourceSystemCode%`	EInvoice source system code
`%eInvoiceGrandTotalAmount%`	EInvoice grand total amount. This is the sum of EInvoice net amount and EInvoice tax amount.
`%folioNumber%`	EInvoice folio number
`%folioPrefix%`	EInvoice folio prefix
`%customerTaxId%`	Customer's tax ID
`%customerName1%`	Customer's name 1 – Possible value: customer's first name
`%customerName2%`	Customer's name 2 – Possible value: customer's second/last name
`%customerCompanyId%`	Customer's company ID
`%customerType%`	Customer type
`%customerAddressSequenceNumber%`	Customer address: sequence number
`%customerAddressStreetName%`	Customer address: street name
`%customerAddressCity%`	Customer address: city
`%customerAddressCountry%`	Customer address: county
`%customerAddressState%`	Customer address: state
`%customerAddressCounty%`	Customer address: country
`%customerAddressPostalCode%`	Customer address: postal code
`%customerAddressStreeNumber%`	Customer address: street number
`%customerAddressApartmentNumber%`	Customer address: apartment number
`%customerAddressNeighborhood%`	Customer address: neighborhood
`%customerAddressReference%`	Customer address: reference
`%customerAddressTelephoneNumber%`	Customer address: telephone number
`%customerAddressEmail%`	Customer address: email

Document Variables	Description
`%customerSourceSystemCode%`	Customer's source system code
`%customerTaxIdPart1%`	Customer's tax ID part 1
`%customerTaxIdPart2%`	Customer's ID part 2
`%customerTaxIdPart3%`	Customer's ID part 3
`%fbItemGrossPriceSum%`	Sum of all food and beverage items gross price
`%fbItemNetPriceSum%`	Sum of food and beverage items net price
`%fbItemExist%`	Flag to indicate that at least one food and beverage item exist
`%storeTaxId%`	Store's tax ID
`%storeName1%`	Store's name 1
`%storeName2%`	Store's name 2
`%storeId%`	Store's ID
`%storeAddressSequenceNumber%`	Store address: sequence number
`%storeAddressStreetName%`	Store address: street name
`%storeAddressCity%`	Store address: city
`%storeAddressCounty%`	Store address: county
`%storeAddressState%`	Store address: state
`%storeAddressCountry%`	Store address: country
`%storeAddressPostalCode%`	Store address: postal code
`%storeAddressStreetNumber%`	Store address: street number
`%storeAddressApartmentNumber%`	Store address: apartment number
`%storeAddressNeighborhood%`	Store address: neighborhood
`%storeAddressReference%`	Store address: reference
`%storeAddressTelephoneNumber%`	Store address: telephone number
`%storeAddressEmail%`	Store address: email
`%storeCompanyTaxId%`	Store Company's tax ID
`%storeCompanyName1%`	Store Company's name 1
`%storeCompanyName2%`	Store Company's name 2
`%storeCompanyId%`	Store Company's ID
`%storeCompanyFiscalRegime%`	Store Company: fiscal regime
`%storeCompanyAddressSequenceNumber%`	Store Company address: sequence number
`%storeCompanyAddressStreetName%`	Store Company address: street name

Document Variables	Description
`%storeCompanyAddressCity%`	Store Company address: city
`%storeCompanyAddressCounty%`	Store Company address: county
`%storeCompanyAddressState%`	Store Company address: state
`%storeCompanyAddressCountry%`	Store Company address: country
`%storeCompanyAddressPostalCode%`	Store Company address: postal code
`%storeCompanyAddressStreetNumber%`	Store Company address: street number
`%storeCompanyAddressApartmentNumber%`	Store Company address: apartment number
`%storeCompanyAddressNeighborhood%`	Store Company address: neighborhood
`%storeCompanyAddressReference%`	Store Company address: reference
`%storeCompanyAddressTelephoneNumber%`	Store Company address: telephone number
`%storeCompanyAddressEmail%`	Store Company address: email

The following variables are only available in the following forms:	ITEM_DATA, REGULAR_ITEMS_DETAIL, FB_ITEMS_DETAIL
`%sku%`	Item SKU
`%department%`	Item department
`%description%`	Item description
`%qty%`	Quantity of the item
`%weighted%`	Item weight
`%uniqueId%`	Unit identifier of item
`%measuringUnit%`	Item unit of measure
`%grossPrice%`	Item gross price
`%netPrice%`	Item net price
`%grossUnitPrice%`	Item gross unit price
`%unitPrice%`	Item unit price
`%itemFolioNumber%`	Item folio number
`%itemFolioPrefix%`	Item folio prefix
`%eligibilityStateCode%`	Item eligibility state code
`%eligibilityStateText%`	Item eligibility state text

The following are only available in TENDER_TYPE form.	
`%tenderId%`	Tender ID
`%tenderDescription%`	Tender description
`%tenderAccountNumber%`	Tender account number. Masked according to eInvoiceConfig.properties' masking.rule property.
`%tenderAmount%`	Tender amount

The following are only available in TAX_TOTALS_DATA form.	
`%taxTypeSubTotalAmount%`	Tax type subtotal amount
`%taxTypeSubTotalTaxInfoTaxId%`	Tax type subtotal tax ID
`%taxTypeSubTotalTaxInfoTaxDesc%`	Tax type subtotal tax description
`%taxTypeSubTotalTaxInfoTaxRate%`	Tax type subtotal tax rate
`%taxTypeSubTotalTaxInfoTaxAmount%`	Tax type subtotal tax amount
The following are only available in `TAXABLE_SUBTOTAL_DATA` form.	
`%taxableSubTotalAmount%`	Taxable subtotal amount. The amount used in this variable is one of the following: If customer tax ID is equal to value specified in tax.generalnational.ids property (eInvoiceConfig.properties), the taxable subtotal amount is recalculated with the sum of the entire items gross price for each tax ID. Otherwise the taxable subtotal amount is the taxable subtotal amount of the current tax type.
`%taxableSubTotalDesc%`	Taxable subtotal tax description.

10.3 List of Generic Variables

Listed below are the available generic variables that can be used in the EInvoice receipt document configuration. The values of these variables are static and constant for all EInvoice generation transactions. These generic variables are mostly from `literals.properties` and some properties are from `eInvoiceConfig.properties` or `eInvoiceFormatConfig.properties`.

Recommendation

All properties in the `literals.properties` are available as generic variables in the EInvoice receipt document configuration. Adding or updating properties in `literals.properties` is not recommended.

Therefore, it is highly recommended to update the `eInvoiceReceiptDocument.form` and literally add the actual text.

Note

The list below may not include all the properties in the `literals.properties`. Only relevant properties are listed.

Generic Variables	Value
`amount_lbl`	Importe
`approval_no_lbl`	N° de aprobación.:
`approval_year_lbl`	Año de aprobación:
`cashier_lbl`	Cajero:
`certicate_no_lbl`	N° de Certificado CSD:
`customer_lbl`	Cliente:
`date_lbl`	Fecha:
`description_lbl`	Descripcion
`digital_seal_lbl`	Sello Digital del SAT:
`fiscal_regime_lbl`	Regimen Fiscal:
`fixed_text_1`	El pago de la contraprestación se efectúa en una sola exhibición.
`fixed_text_2`	Este documento es una representación impresa de un CFD.
`folio_no_lbl`	Folio Interno: No.
`issue_place_date_lbl`	Fecha y hora de emisión:
`original_message_lbl`	Cadena original del complemento de certificación digital del SAT:
`quantity_lbl`	Cantidad
`sat_certificate_no_lbl`	N° de Certificado SAT:
`sat_digital_seal_lb`	Sello Digital del CFDI:
`stamp_date_lb`	Fecha y hora de Certificación:
`store_fmt_lbl`	Tienda:
`subtotal_general_lbl`	Sub-total general
`subtotal_lbl`	Subtotal
`taxID_lbl`	RFC:
`telephone_lbl`	Tel.
`tender_acct_num_lbl`	No. de Cta:
`tender_desc_lbl`	Forma de Pago:
`uuid_lbl`	Folio Fiscal:

Generic Variables	Value
terminal_lbl	Terminal:
time_lbl	Hora:
total_lbl	Total
trx_lbl	Trns:
unit_price_lbl	P. Unitario
zipcode_lbl	C.P.

The list below is generic variables that are from eInvoiceConfig.properties:

Generic Variables	Description
TOTAL_LINE_CHAR	Value of total_line_char property of eInvoiceFormatConfig.properties.
CURRENCY_SYMBOL	Value of currency.sign property of eInvoiceConfig.properties.
EINVOICE_SRC_SYSTEM_CODE	Value of eInvoice.srcSystemCode property of eInvoiceConfig.properties.
EINVOICE_TRX_ORIGIN_CODE	Value of eInvoice.trxOriginCode property of eInvoiceConfig.properties.
EINVOICE_CONSUMO_ITEM_SKU	Value of eInvoice.consumoItemSku property of eInvoiceConfig.properties.
EINVOICE_CONSUMO_UNIT_OF_MEASURE	Value of eInvoice.consumoUnitOfMeasure property of eInvoiceConfig.properties.
TAX_EXEMPT_DESCRIPTION	Value of tax.exempt.description property of eInvoiceConfig.properties.
GENERAL_NATIONAL_TAX_IDS	Value of tax.generalnational.ids property of eInvoiceConfig.properties.
GENERAL_FOREIGN_TAX_IDS	Value of tax.generalforeign.ids property of eInvoiceConfig.properties.
COMMA	Comma (,) character

10.4 EInvoice 2D Barcode Support

Refer to SAP Note: 1954430 - SAP POS 2.3 Receipt Preprocessor Technical Reference for details about how to add 2D barcode support on the EInvoice receipt.

www.sap.com/contactsap