ERP-727 Integration of EDICOM with S4 VIM to Automate e-invoicing

Status	Approved
Owner	SHARMA-ext, Meenakshi
Stakeholders	Laurence DANNET, Paulo Golinelli
Jira Request ID	ERP-135 - Jira project doesn't exist or you don't have permission to view it.
Jira Development ID	ERP-727 - Jira project doesn't exist or you don't have permission to view it.

High- Level Specification

Parameter	Value
Application System	EDICOM,S/4HANA ROW, S/4HANA China
Business Process Reference	03.04.06.01. Manage Invoice Receipt

Functional Overview

This interface supports the automated receipt of electronic invoices from **EDICOM Middleware** into **OpenText SAP VIM** via web services. Its purpose is to enable a streamlined, touchless invoice intake process and initiate the **VIM Document Processing (DP) workflow**.

Scope and Objectives

The objective of this functional specification is to define an interface that receives vendor invoice registered on the E-Invoice portal. The invoices are fetched by EDICOM and transmitted to SAP Vendor Invoice Management (VIM) in XML format via an inbound proxy. The interface ensures seamless, accurate, and secure transfer of invoice data into VIM for further processing.

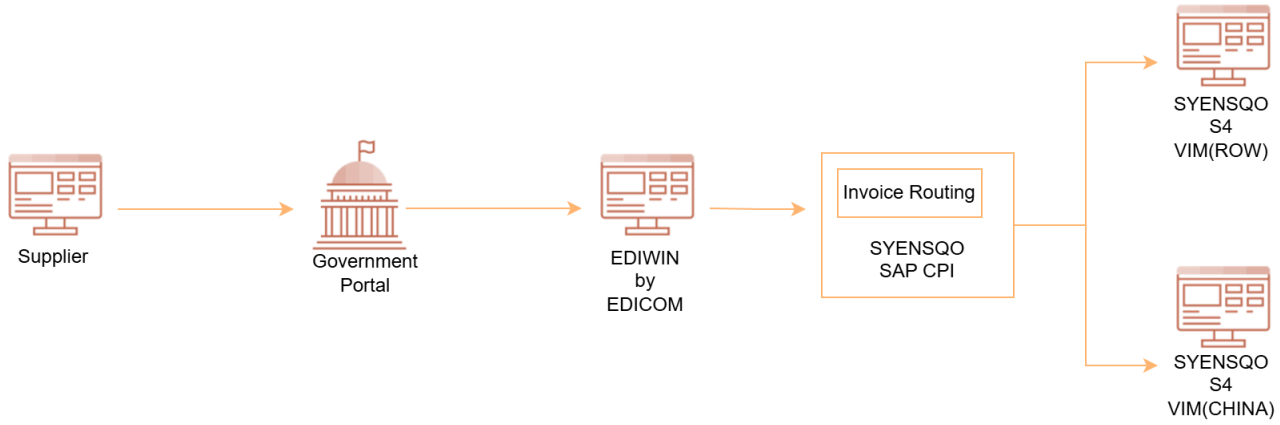
Scope

- Capture vendor invoices registered on the E-Invoice portal and retrieved by EDICOM
- Receive invoice data in SAP VIM through an inbound proxy interface in XML format
- Validate the incoming XML data for structure, completeness, and mandatory fields
- Create and register invoices in SAP VIM based on the received data
- Handle error logging, monitoring, and exception management for failed or incomplete messages
- Support standard invoice scenarios

Out of Scope:

- Manual invoice entry in SAP
- Invoice approval workflows within VIM
- Downstream FI posting and payment processing

Process Flow Diagram



Step	Description	Comment
Step 1	The supplier is responsible for registering the invoices on the E-Invoice portal.	
Step 2	EDIWIN tool by EDICOM will extract Syensqo vendor invoices from the E-Invoice portal.	
Step 3	EDIWIN will publish the XML file in the country-specific e-invoice format supported by VIM.	For example Italy XMLs will be in FATTURA format
Step 4	SAP CPI(Cloud Platform Integration) system will be middleware between EDIWIN by EDICOM and S/4 VIM and responsible to route the incoming e-invoice to right S/4 instance	For example: If China e-invoices will be in scope it has to be routed to China S/4 VIM instance. Rest of the world invoices should be routed to ROW S/4 VIM Instance.
Step 5	A proxy interface will be established for receiving e-invoices.	Inbound Proxy file will have country code and invoice payload in base 64 format.
Step 6	An application job class will be implemented that calls the proxy to check for new invoices and register the incoming XML under the country-specific VIM e-invoice XML channel and archive ID.	
Step 7	*While registering the invoice in VIM, in the event of a connection issue with the archive server for the content repository, a Launchpad notifications will be triggered.* *Table will be maintained for recipient groups for the notifications in case of error.*	
Step 8	*Further enhancements to XML parsing will be implemented based on country-specific e-invoice requirements.*	
Step 9	*XML fields will be mapped to the VIM header and item tables as per the mapping of semantic names of electronic document in Invoice Inbound configuration.*	It will done as part of configuration

Assumptions

EDIWIN will transmit invoices to VIM via the inbound proxy and will route only Syensqo vendor invoices that are designated for VIM processing. EDICOM will be routing the invoice to correct S4 VIM instance.

One invoice per XML.

Dependencies

This enhancement is dependent on the capabilities and interfaces to be agreed upon with EDICOM. It is also subject to the legal requirements of the countries where vendor e-invoicing is applicable.

It is dependent on ERP-730.

Security, Integrity and Controls

Inbound Proxy: The inbound proxy file will contain the invoice payload encoded in Base64 format.

Control Table: A custom table will be maintained to store the country, company code, and transaction type values that are in scope for e-invoicing.

Configuration Requirements

- Configure input channel **OAWD_XXXX** in SAP VIM for each Syensqo E-invoicing country to receive XML invoices from the EDICOM system.
- Configure the archive document type to process XML invoices and enable visualization/rendering of XML content for each Syensqo E-invoicing country.
- Link country-specific standard module handlers to the corresponding e-invoice channels.
- Based on Syensqo requirements, configure an enhanced XML parsing class.
- XML fields will be mapped to VIM header and item fields

Language Requirements

No specific language requirement for the integration. Error messages to be logged in English.

Special Requirements

An application job class will be implemented in the inbound proxy to register the incoming XML under the dedicated XML channel and archive ID corresponding to the relevant country and company code.

In the event of a connection issue with the archive server for the related content repository, a launchpad notification will be triggered and sent to the identified recipient groups.

A table will be used to store the recipient groups, who will be notified in case of such failures. Once the archive server issue is resolved, the team can access /AIF/IFMONI to review and clear the error messages.

Design Rationale

Functional Requirements

SAP VIM will receive e-invoices from EDICOM through an inbound proxy interface in a country-specific XML format. The inbound message will contain the country and company code along with the invoice payload encoded in Base64 format. The country and company code combination will be used to correctly identify and route the invoice to the appropriate VIM input channel for further processing.

Proposed Technology to Use

Inbound proxy and custom class to register the invoices in VIM will be used.

Data Source Considerations

N/A

Table	Field Name	Comments/Calculation/Field Manipulation

Data Validation Considerations

N/A

Table	Field Name	Comments/Calculation/Field Manipulation

Custom Tables

N/A

Master Data

N/A

Field	Description	Data Type/Length	Validation rule/ Value Help

Configuration Table

N/A

Field	Description	Data Type/Length	Validation rule/ Value Help

Selection Screen Enhancement

N/A

Field Name	Description	Select:	Data Type/Length	Default Value/ Validation rule/ Value Help	Selection Logic

Processing Logic

E-Invoice Registration Enhancement

1. **Inbound XML Receipt**
    - XML invoices will be received via the inbound proxy.
    - An enhancement will be triggered during inbound processing to handle XML invoice registration in VIM.
2. **Country and Content Repository Determination**
    - During registration, the enhancement reads the country and XML format from the XML.
    - Based on the country and XML format, the Content Repository and country-specific VIM e-invoice XML channel details will be identified for the XML and if there is any attachment should be stored.
3. **VIM Registration**
    - The XML invoice is registered in VIM with the determined Content Repository and country-specific VIM e-invoice XML channel.
    - The original XML file is stored as an attachment against the VIM document.
    - Additional attachments will also be retrieved and stored
4. **Content Server Connectivity Check**
    - While registering the document, the enhancement checks the Content Server connection.
    - If the Content Server connection fails:
        - The registration process is stopped.
        - An error is logged in AIF.
        - Send out AIF alerts to the AP support team as soon as registration fails.
5. **Reprocessing Mechanism**

- Once the Content Server issue is resolved:
    - The error entry in AIF is manually cleared.
    - The message is reprocessed.
    - The XML invoice is registered successfully in VIM without requiring retransmission from the source system.

Volumetrics

Volume is driven by the number of invoices that we receive through proxy.

Performance Considerations

No specific performance considerations since we just receive the invoices.

Error Handling

Invoice registration interface failures can be monitored through Launchpad notifications as well as via transaction **/AIF/IFMON**

Testing

How to Test

Based on interface with EDICOM, E-invoice files will be received in VIM and documents will be registered in S4 VIM and workflow will be triggered with XML data mapped to VIM header and item fields.

Test Conditions and Expected Results

ID	Condition	Expected Result
1	Send XML invoice from EDICOM to VIM	DP gets registered in VIM and XML gets archived in respective country content repository.
2	Send XML invoice from EDICOM to VIM while the archive server is down	DP does not get registered as archive server is down. And launchpad notification will be triggered to identified recipients.
3	Send XML invoice embedded with PDF attachment from EDICOM to VIM	DP gets registered in VIM and XML & attachments gets archived in respective country content repository.
4	Send XML invoice from EDICOM to VIM	DP gets registered in VIM as per the values received from source XML nodes.
5	Send XML invoice with missing mandatory nodes to VIM	DP not created, error logged.
6	Send XML invoices for different countries.	DP gets registered under correct company code and archived under correct country specific content repository
7	Send an XML invoice with a large embedded PDF	Successful processing within system limits or controlled error handling.
8	Clear **/AIF/IFMON** error and process the error files	DP gets registered and archived successfully as long as archive server is up and there is no other issues
9	Send duplicate XML invoice from EDICOM	Duplicate invoice is detected and handled as per legal requirement to handle duplicate e-invoices of respective country
10	Send XML invoice with invalid XML structure	Invoice is rejected with appropriate error logging and notification as designed as part of EDICOM interface
11	Send XML invoice with special characters / encoding differences	Invoice is processed correctly without data corruption

Test Considerations/Dependencies

Other Information

Development Details

Package

Package Name	Parent Package

Enhancement Implementation

Enhancement Type	Standard Definition Name	Custom Implementation Name	Design Rationale Reference

Other Development Objects

Object Type	Object Name	Purpose/High Level Logic	Design Rationale Reference

Appendix

Custom Authorization Group Naming Convention

This table is based on the Syensqo development standards document. It provides the naming conventions for authorization groups to associated with custom reports and tables to comply with security requirements.

ABAP	ZFI	ZMM	ZPS	ZCO	ZSD	ZBC	ZFI	ZCA
TABLES	ZFIT	ZMMT	ZPST	ZCOT	ZSDT	ZBCT	ZFIT	ZCAT

See also

File	Modified
File drawio-backup-EDICOM-VIM Integration-rev-8 draw.io diagram backup	Feb 27, 2026 by JOUHAUD-ext, Yoann
File EDICOM-VIM Integration draw.io diagram	Feb 19, 2026 by SHARMA-ext, Meenakshi
File ~EDICOM-VIM Integration.tmp draw.io Draft	Feb 19, 2026 by SHARMA-ext, Meenakshi

Download All

Change log

Version	Published	Changed By	Comment
CURRENT (v. 37)	**Apr 28, 2026 13:58**	**WEINERT-ext, Patrick**	
v. 36	Apr 28, 2026 13:52	**SHARMA-ext, Meenakshi**	
v. 35	Apr 28, 2026 13:37	**SHARMA-ext, Meenakshi**	
v. 34	Apr 28, 2026 13:30	**SHARMA-ext, Meenakshi**	
v. 33	Apr 28, 2026 10:59	**SHARMA-ext, Meenakshi**	
v. 32	Mar 03, 2026 10:29	**WEINERT-ext, Patrick**	
v. 31	Feb 24, 2026 13:25	**SHARMA-ext, Meenakshi**	
v. 30	Feb 23, 2026 09:40	**SHARMA-ext, Meenakshi**	
v. 29	Feb 20, 2026 15:48	**SHARMA-ext, Meenakshi**	
v. 28	Feb 20, 2026 15:42	**SHARMA-ext, Meenakshi**	

Go to Page History