



SAP Logistics Business Network, Global Track and Trace Option Track Purchase Orders - Deep Dive with SAP ERP Integration

Logistics Business Network
December 2020

PUBLIC

Objectives



After completing this learning module, you will be able to:

- Learn what prerequisite is necessary for Global Track and Trace Option
- Learn how to maintain IDOC configurations in ERP for integration
- Learn how to maintain extractors in ERP for integration
- Learn how to download and implement sample ABAP codes from Github
- Learn how to customize own logic based on sample codes

Agenda

A Prerequisites

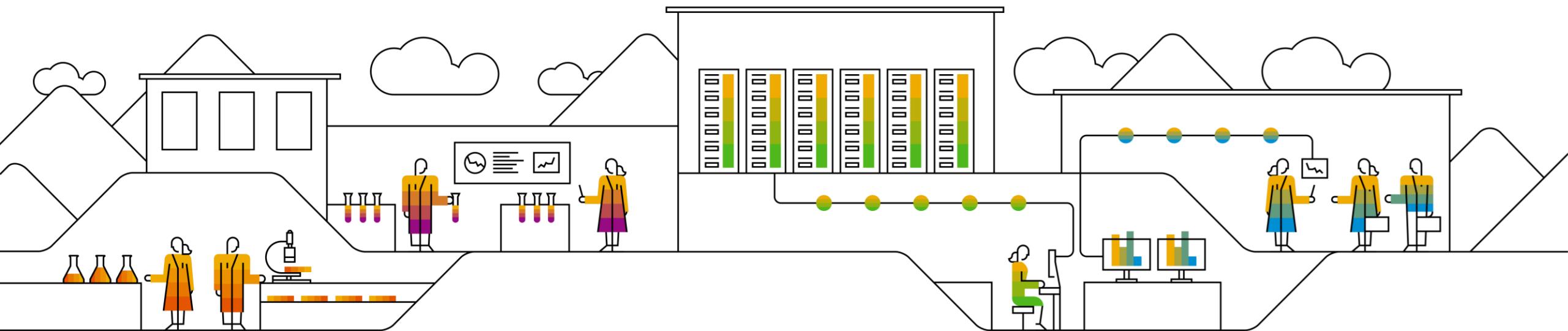
B Configuration and Implementation - Basic

 B1 IDOC Configuration

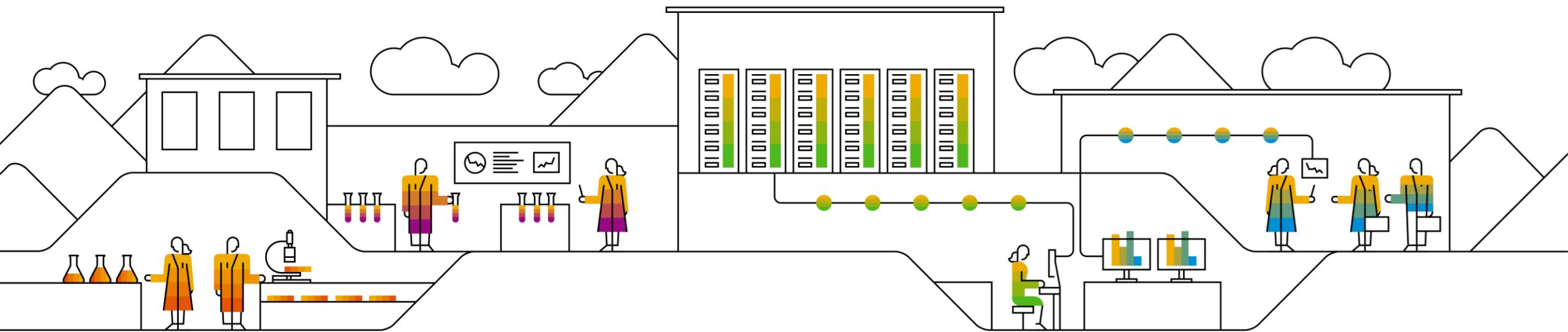
 B2 Extractor Configuration

C Download ABAP Code from GitHub

D Configuration and Coding Guide - Advanced



A) Prerequisites



STEP 1: Check the SAP Version

- 1-1: The SAP Product Version for GTT v2 shall be SAP EHP1 FOR SAP NETWEAVER 7.3 or higher.
- 1-2: SAP NOTE 2937175 shall be implemented.
- 1-3: The ABAP codes to support sample apps for GTT v2 can be implemented in S4 HANA 2101 on premise, which is not validated in lower release.

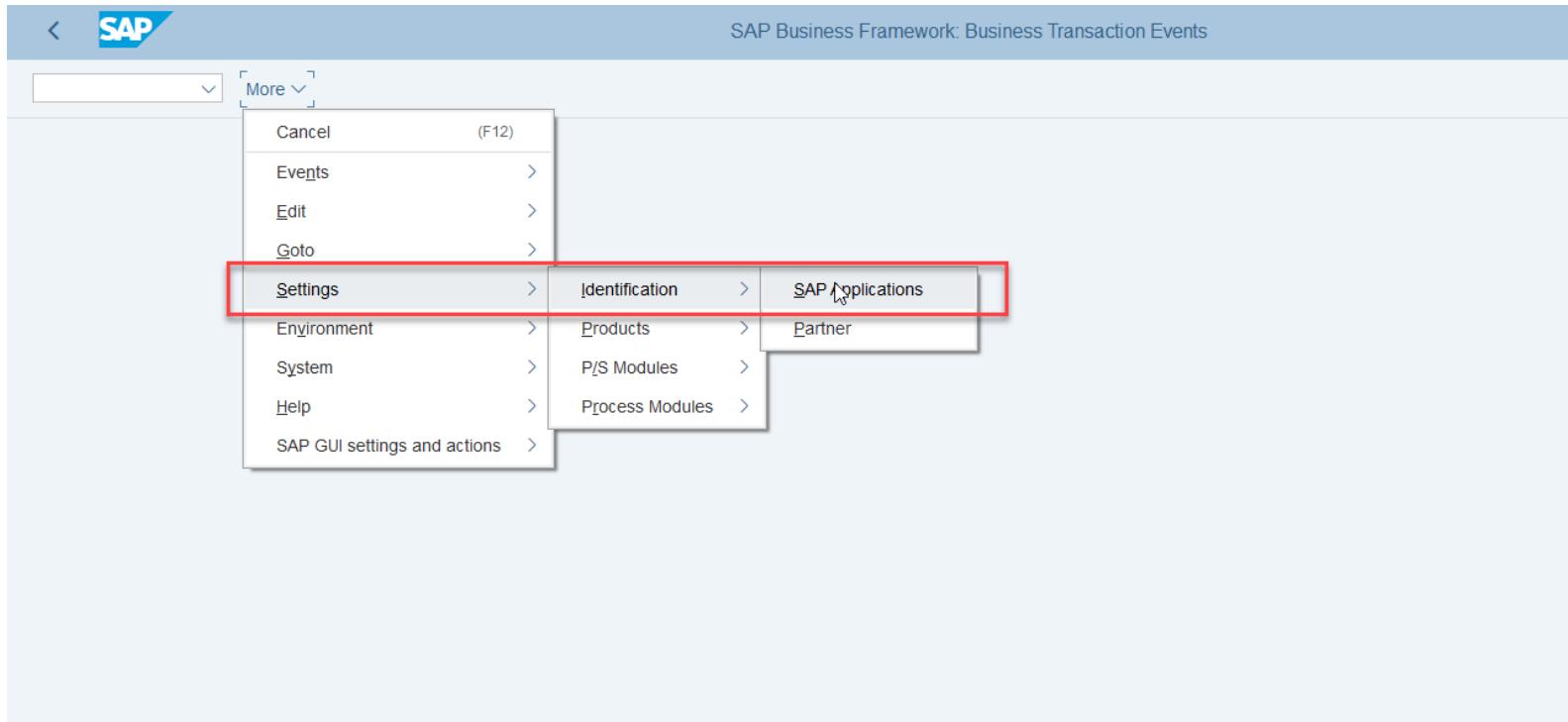
TIPs:

- 1, SAP version reference: <https://support.sap.com/en/my-support/software-downloads/support-package-stacks/product-versions.html#section>
- 2, Note-assistant reference: <https://support.sap.com/en/my-support/knowledge-base/note-assistant.html>

SAPNotes								
11 SAP Note(s) found								
SAP Component	Number	Versi...	Score	Title	Changed On	Status	Responsible	Category
SCM-EM-AS	2959576	1	1	Amendments to EM API for LBNTT2.0	18.08.2020	In Process	Thomas Rumbach	Program error
SCM-EM-AS	2937175	1	1	Enhancement of IDOCs sent to GTT	16.09.2020	Released for Customer	Thomas Rumbach	Advance development
SCM-EM-AS	2834393	1	1	Solving ATC Issues	27.09.2019	Released for Customer	D046164	Program error
SCM-EM-AS	2819787	1	1	TM-EM integration - analyzing errors	25.07.2019	In Process	Bernd Sieger	Help for error analysis
SCM-EM-AS-CNF	2798670	1	1	IMG activity inactive: Define SAP EM Extraction Functions	29.05.2019	Released for Customer	Bernd Sieger	Program error
SCM-EM-AS	2609449	4	1	Delete orphaned entries in table /SAPTRX/AOTREF (2)	11.07.2019	Pilot Release	Bernd Sieger	Workaround of missing
SCM-EM-AS	2502086	2	1	Aligning the BAPI processing mode with the communication mode	11.07.2017	Pilot Release	Bernd Sieger	Special development
SCM-EM-AS	2339984	2	1	Orphaned EM inbound queues in application systems	18.04.2019	Released for Customer	Bernd Sieger	Consulting
SCM-EM-AS	2159436	1	1	Runtime-Error "ABAP Programming" when trying to save delivery. System QSC-800	22.04.2015	In Process	D025889	Program error
SCM-EM-AS	1507998	4	1	Expert Consulting in the area of SAP Event Management	09.05.2011	Released for Customer	Florian Frey	Consulting
IS-R-PUR-PCC	896191	3	1	FAQ: EM seasonal procurement (Consulting, Tips, Customizing)	13.07.2006	Released for Customer	Andreas Lange	FAQ

STEP 2: Log on the Development Client to Configure BTE

- 2-1: Ensure you have development access to the client for cross-client customizing and local development
- 2-2: Log on to the client and enter transaction code (T-code): **FIBF**
- 2-3: Click **More -> Settings -> Identification -> SAP Applications**



STEP 2: Activate SAP Event Manager Integration

2-4: Position on the Application ID: **PI-EM**

2-5: Check the field **Application Active**

2-6: Click **Save**

The screenshot shows a SAP Fiori application interface titled "Change View "BTE Application Indicator": Overview". The table lists various SAP applications (Appl.) and their corresponding texts. The application "PI-EM" is highlighted with a red box around its row, and its "Active" checkbox is checked. Other applications listed include PM, PM-BW, PM-EQM, PM-PAM, PM-PC, PMAT, PMIPUR, PMPUSH, PP-BD, PP-DD, PP-MRP, PRICAT, PS-REP, PSRV, QBEXT, QBEXTP, QILPO, RDSVFI, and RDSVMD. The "Active" column contains checkboxes for each row.

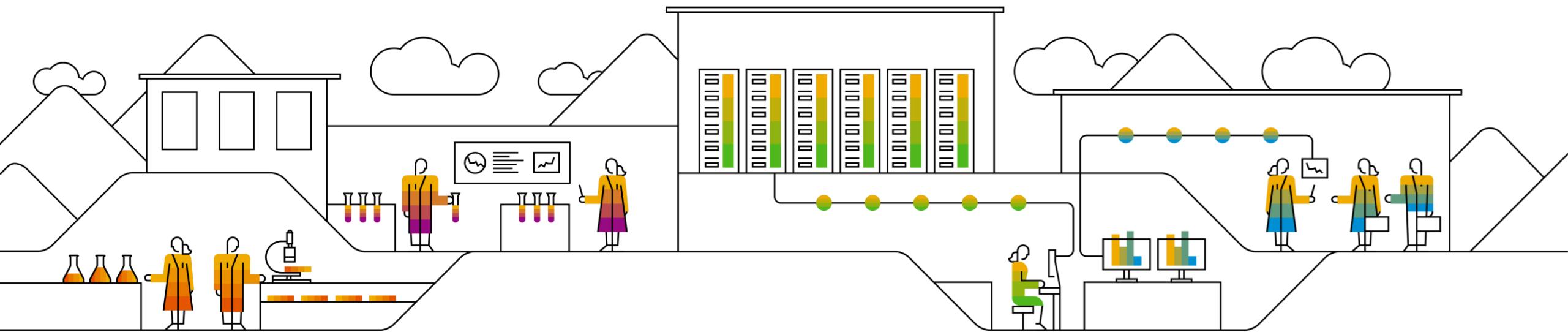
Appl.	A	Text
PI-EM	<input checked="" type="checkbox"/>	SAP Event Manager Integration
PM	<input checked="" type="checkbox"/>	Instandhaltung
PM-BW	<input checked="" type="checkbox"/>	Instandhaltung-BW
PM-EQM	<input checked="" type="checkbox"/>	Instandhaltung, Equipment
PM-PAM	<input checked="" type="checkbox"/>	Instandhalt. Pool Asset Mgmt
PM-PC	<input checked="" type="checkbox"/>	Product Compliance
PMAT	<input checked="" type="checkbox"/>	Produkt - Material
PMIPUR	<input type="checkbox"/>	PMI Anschluss Einkauf
PMPUSH	<input type="checkbox"/>	MAM Push
PP-BD	<input checked="" type="checkbox"/>	Production Planning MasterData
PP-DD	<input checked="" type="checkbox"/>	Demand Driven Replenishment
PP-MRP	<input checked="" type="checkbox"/>	Material Requirements Planning
PRICAT	<input type="checkbox"/>	Preiskatalog
PS-REP	<input checked="" type="checkbox"/>	Projektsystem
PSRV	<input checked="" type="checkbox"/>	Produkt - Service
QBEXT	<input checked="" type="checkbox"/>	External Inspection Procurement
QBEXTP	<input checked="" type="checkbox"/>	External Inspection Production
QILPO	<input checked="" type="checkbox"/>	Inspection Lot Order Integr.
RDSVFI	<input type="checkbox"/>	Dgtl.Signature Validation FI
RDSVMD	<input checked="" type="checkbox"/>	Dgtl.Signature BP Check

Buttons at the bottom: "Position...", "Entry 133 of 174", "Save", "Cancel".

B) Configuration and Implementation

- Basic

B1. IDOC Configuration



STEP 1: Define RFC Connection for GTT

1-1: Log on to the business client

1-2: Enter T-code **SPRO** and then click **SAP Reference IMG** to open **Display IMG** page

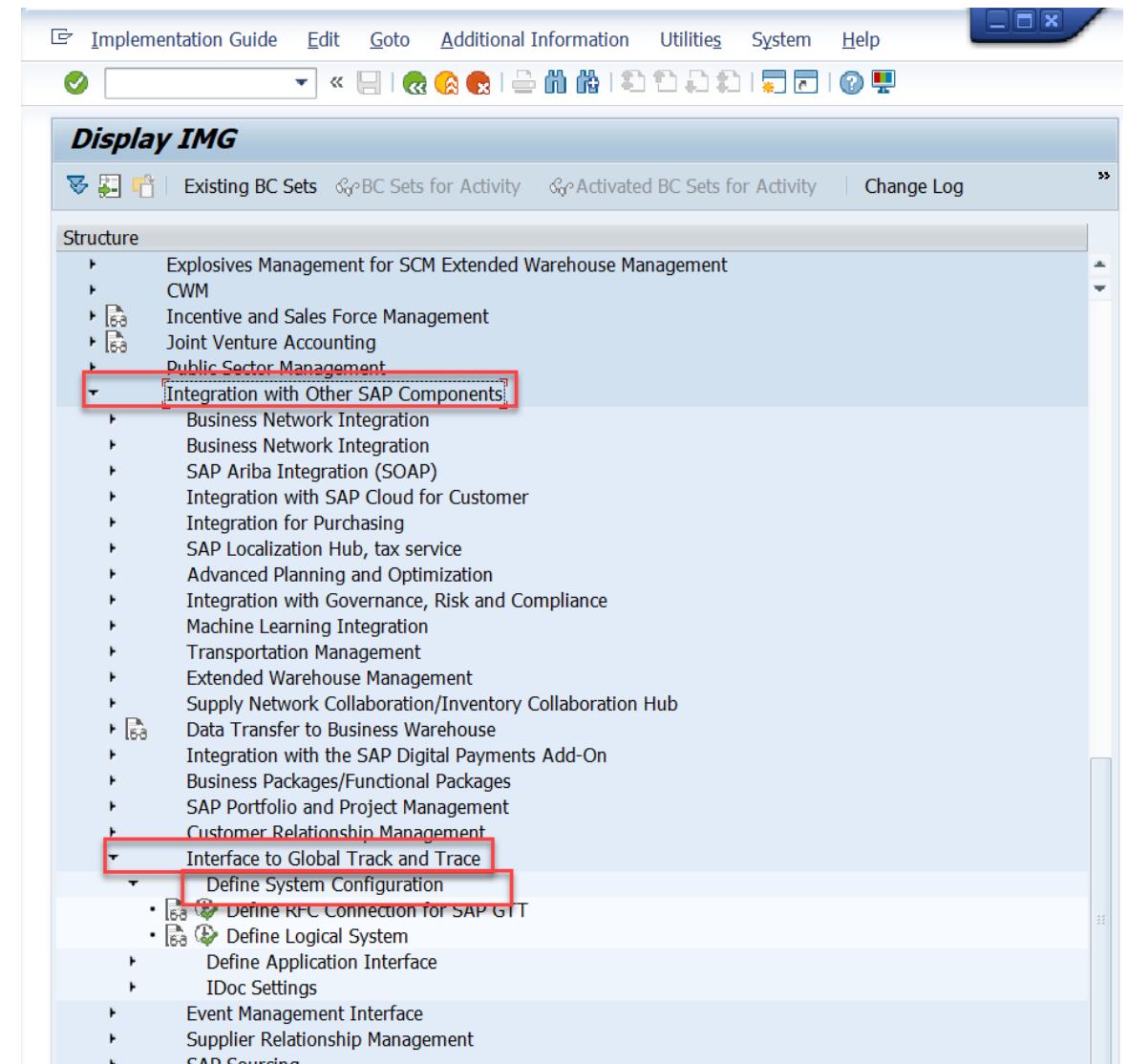
1-3: Click **Integration with Other SAP Components**

-> **Interface to Global Track and Trace**

-> **Define System Configuration**

1-4: Choose activity:

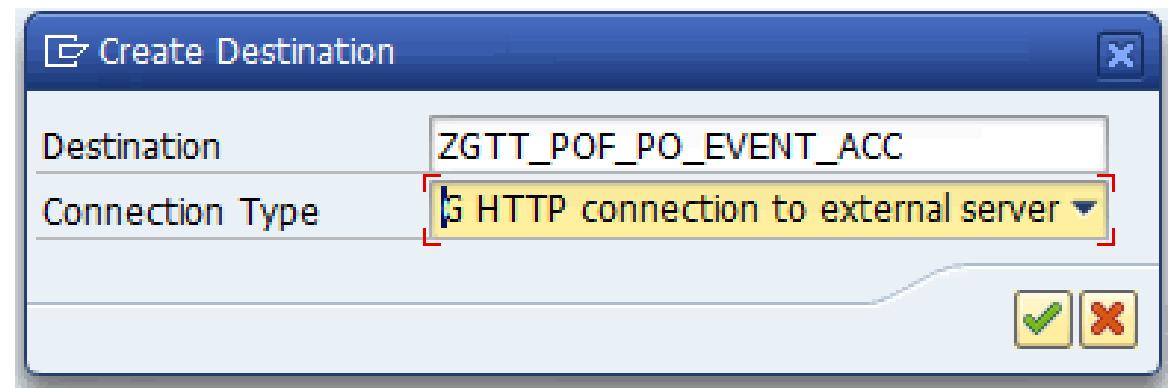
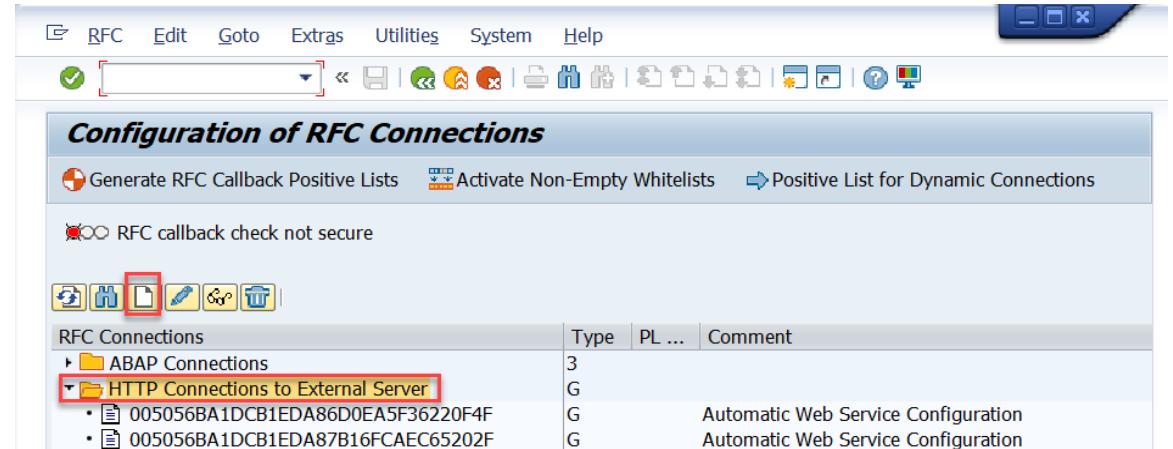
Define RFC Connection for SAP GTT



STEP 1: Define RFC Connection for GTT

1-5: Choose **HTTP Connections to External Server**, click **Create** and create a new RFC connection.

1-6: Fill in the **Destination** and choose the **Connection Type**:
'G-HTTP connection to external server'.



STEP 1: Define RFC Connection for GTT

1-7: Enter a description

1-8: In the **Technical Settings** tab, fill in the **Host, Port and Path Prefix**

For example, the url of solution owners is as below:

<https://sat-so-01.gtt-flp-lbnplatform-pre-live.cfapps.eu10.hana.ondemand.com/>

Host: sat-so-01.gtt-flp-lbnplatform-pre-live.cfapps.eu10.hana.ondemand.com

Port: 443

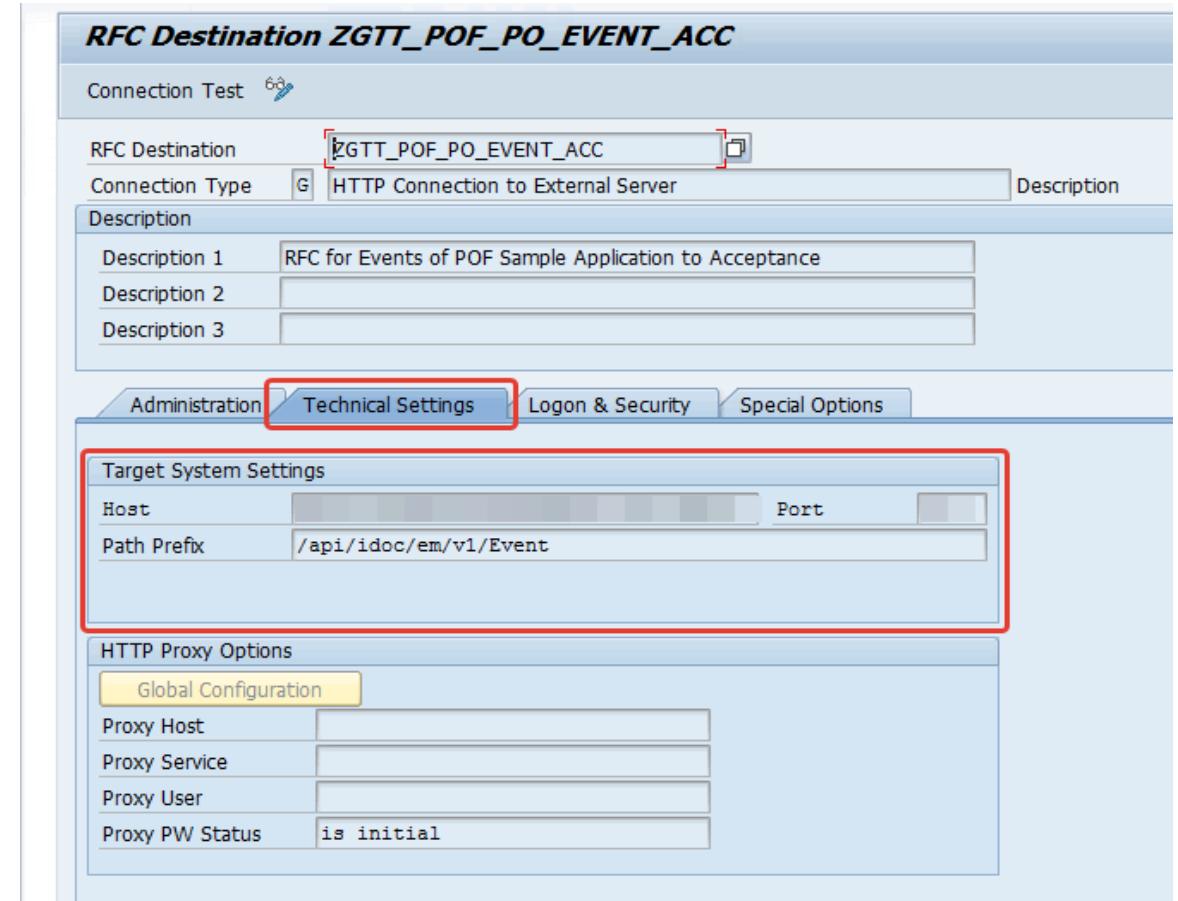
You need to configure two RFC connections separately for event and tracked process. They have different **Path Prefixes**.

For the event:

Path Prefix: /api/idoc/em/v1/Event

For the tracked Process:

Path Prefix: /api/idoc/em/v1/TrackedProcess



STEP 1: Define RFC Connection for GTT

1-9: In the **Logon & Security** tab, enter the Logon information.

For basic authentication, the GTT technical user / password is needed. You can get this from your GTT administrator.

Also, SSL must be *Active*.

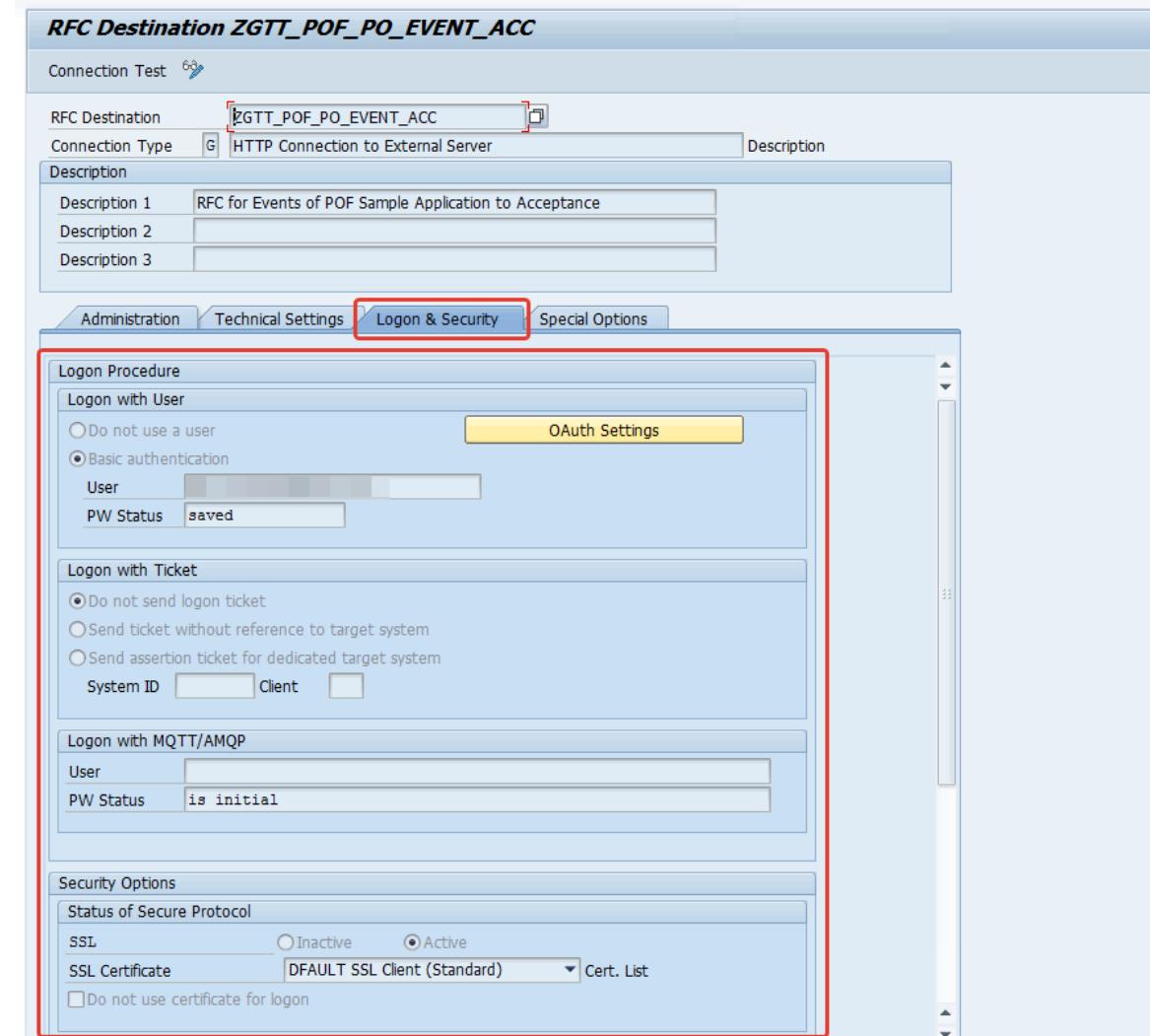
The recommended SSL Certificate is: *DEFAULT SSL Client (Standard)*.

1-10: Save the configuration

1-11: Click **Connection Test**. A successful connection returns a status HTTP response of 200.

Caution: You need to configure two RFC Connections:

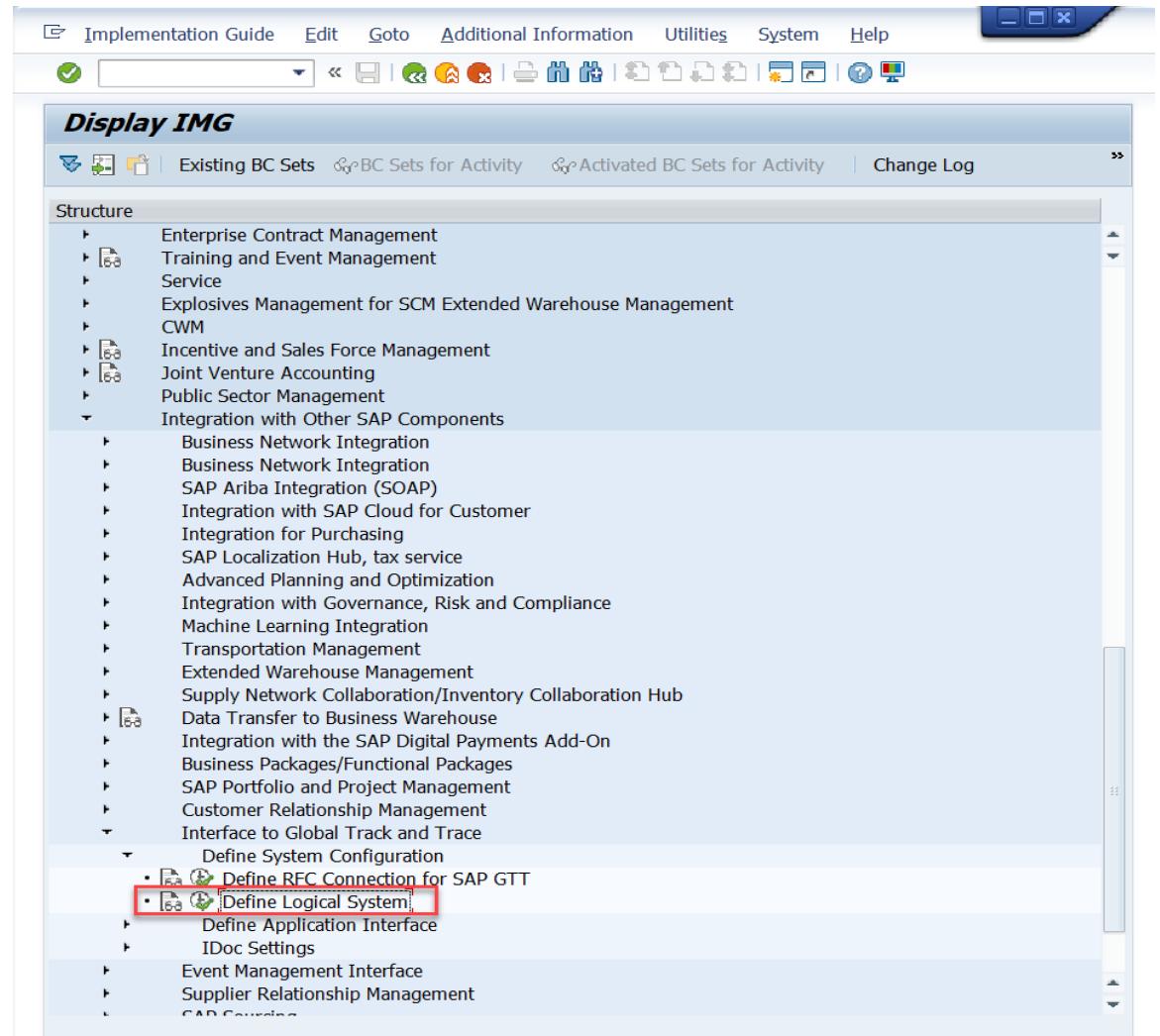
- one for event and
- the other for tracked process.



STEP 2: Define Logical System

2-1: In **Display IMG** page, click **Integration with Other SAP Components** -> **Interface to Global Track and Trace** -> **Define System Configuration**.

2-2: Choose activity **Define Logical System**.

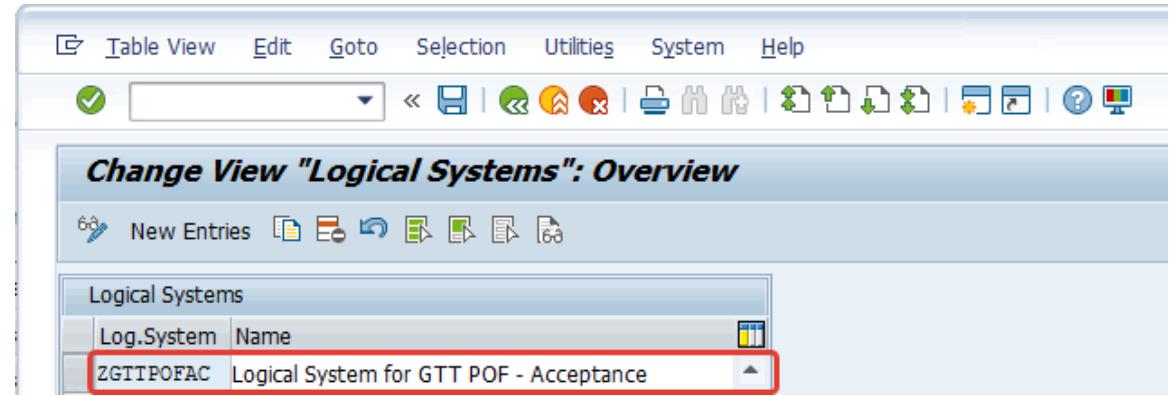


STEP 2: Define Logical System

2-3: Create **New Entries** to create a new Logical System, fill in the:

- Logical system code and
- Name of the new logical system

2-4: Save the configuration



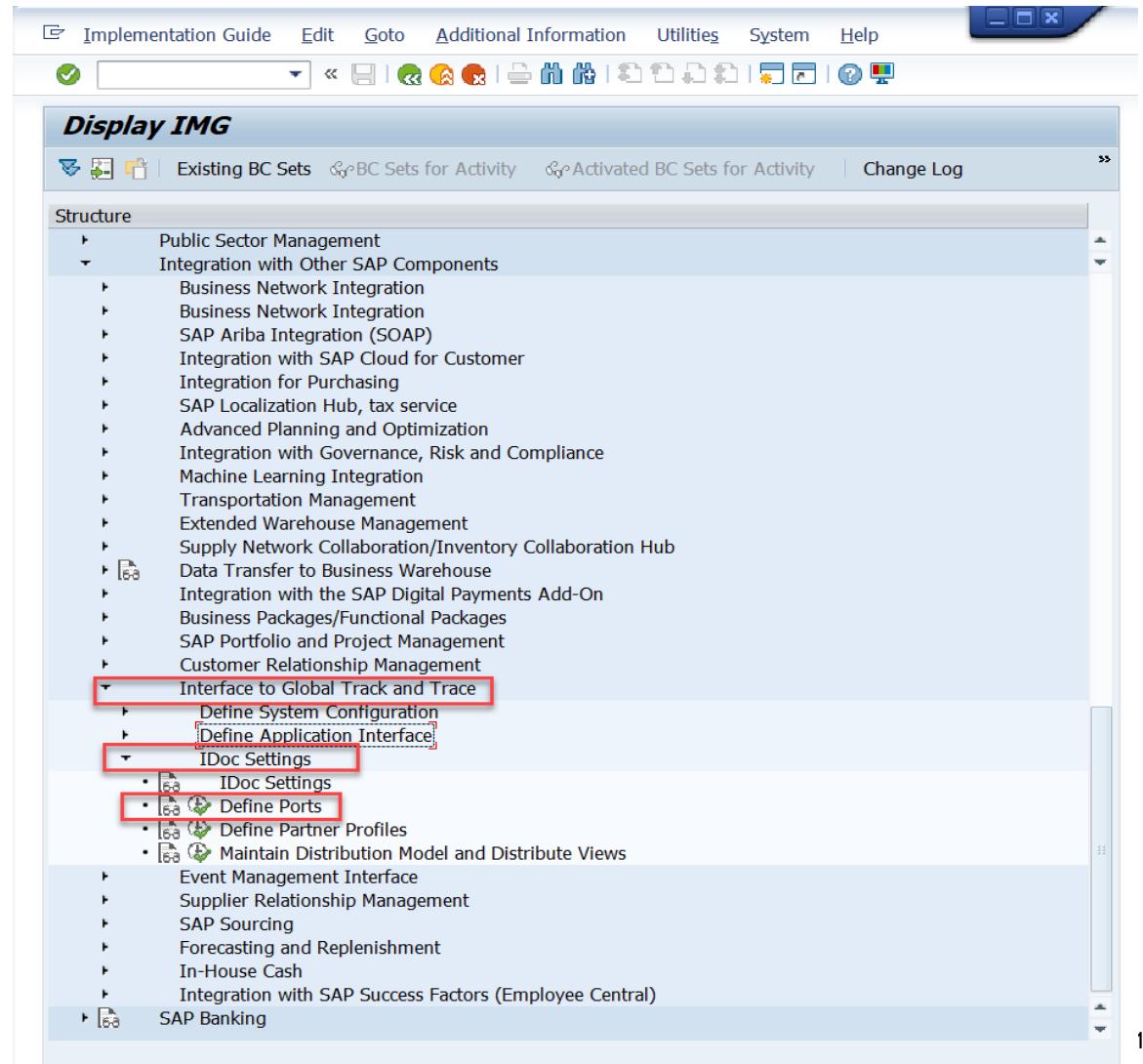
The screenshot shows the SAP GUI interface for defining logical systems. The title bar reads "Change View 'Logical Systems': Overview". Below the toolbar, there is a menu bar with "Table View", "Edit", "Goto", "Selection", "Utilities", "System", and "Help". The main area displays a table titled "Logical Systems" with two columns: "Log.System" and "Name". A single row is visible, containing "ZGTTPOFAC" in the Log.System column and "Logical System for GTT POF - Acceptance" in the Name column. This row is highlighted with a red border.

Log.System	Name
ZGTTPOFAC	Logical System for GTT POF - Acceptance

STEP 3: Define Ports

3-1: In **Display IMG** page, click
Integration with Other SAP Components ->
Interface to Global Track and Trace ->
IDoc Settings

3-2: Choose activity **Define Ports**



STEP 3: Define Ports

3-3: Choose **XML HTTP** folder, and click **Create** to create a new port

3-4: Fill in the **RFC Destination**, it is the RFC connection you created in STEP 1

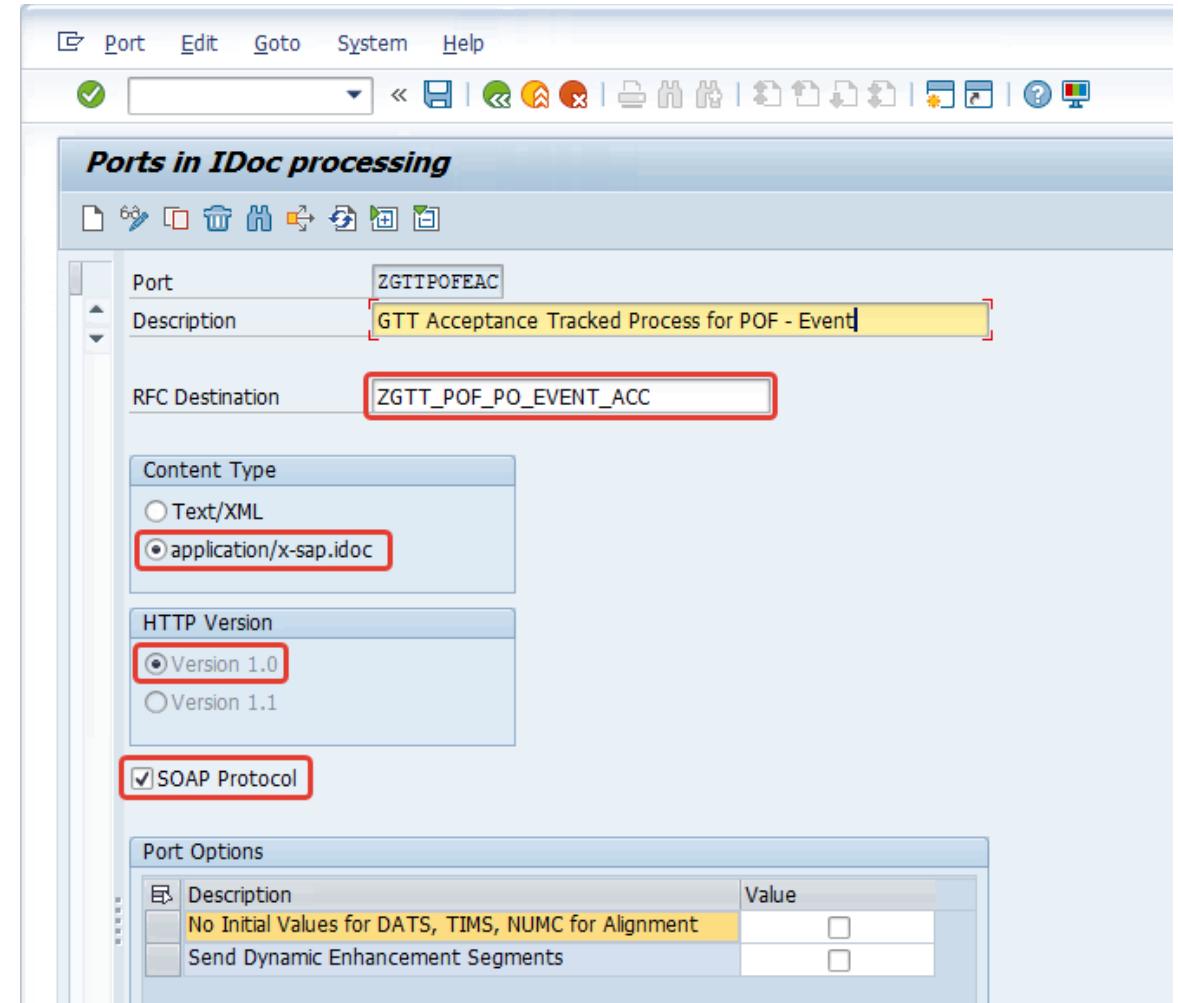
3-5: Choose **Content Type** as *application/x-sap.idoc*

3-6: Choose **HTTP Version** as *Version 1.0*

3-7: Mark it as SOAP Protocol

3-8: Save the configuration

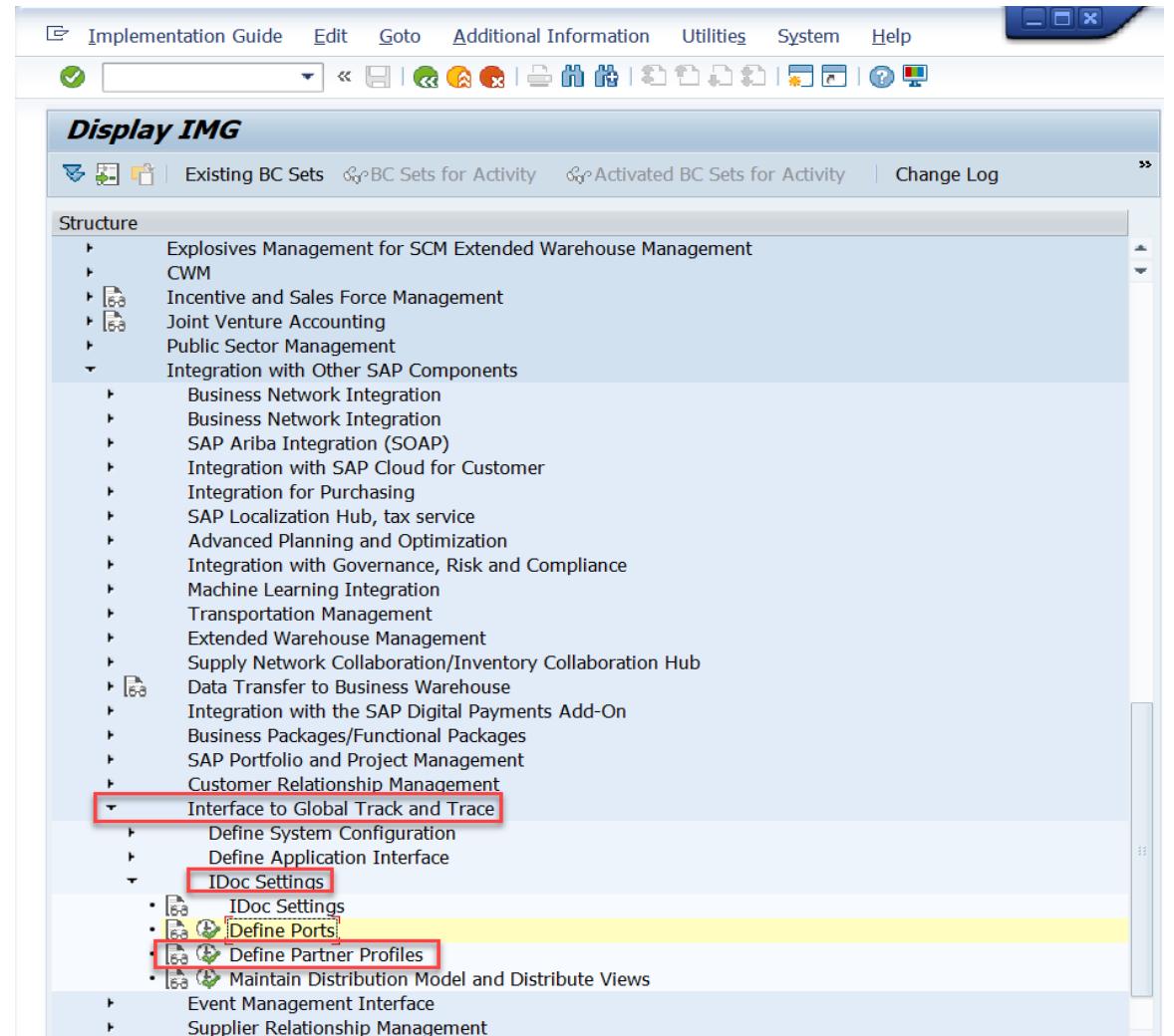
Caution: You need to define two ports, one for event and the other for tracked process.



STEP 4: Define Partner Profiles

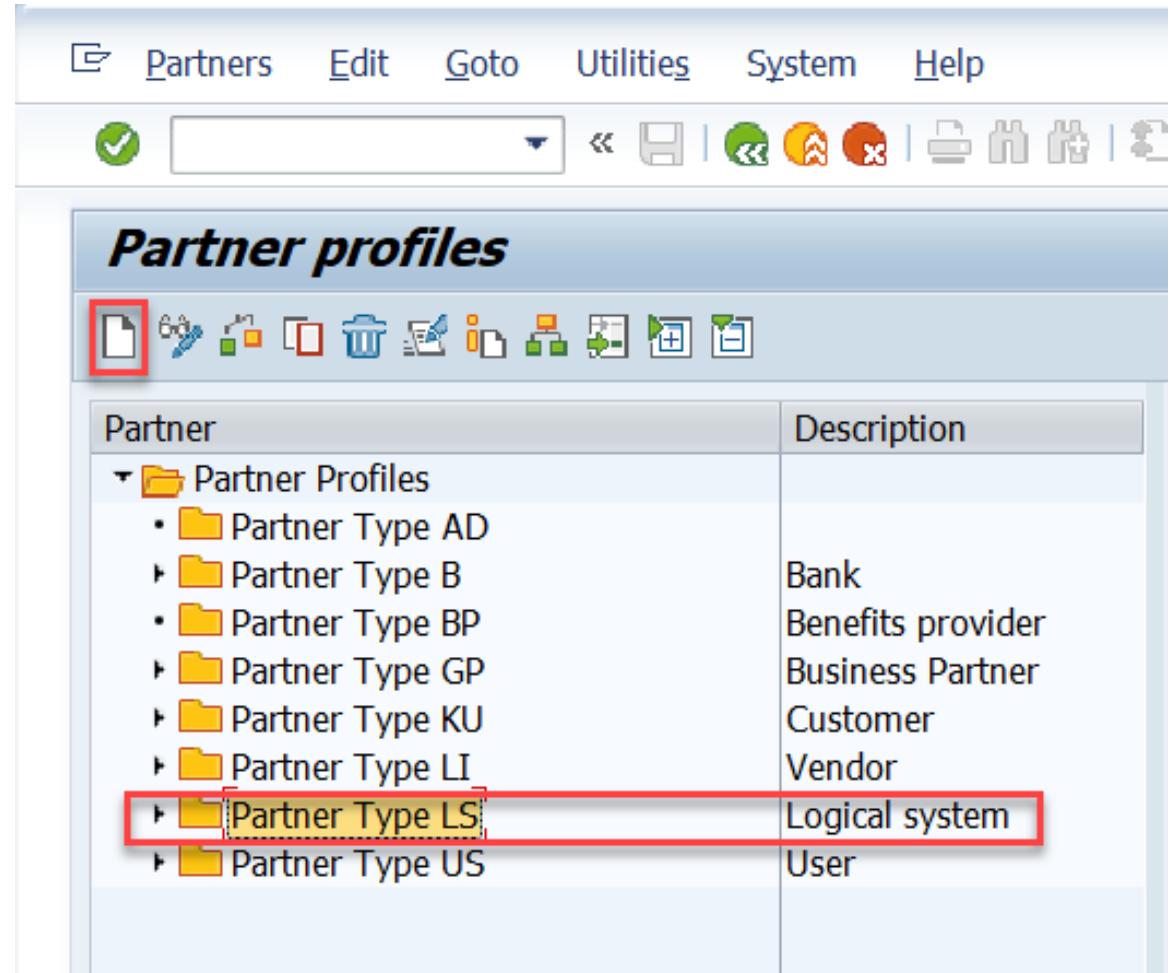
4-1: In **Display IMG** page, unfold **Integration with Other SAP Components** -> **Interface to Global Track and Trace** -> **IDoc Settings**

4-2: Choose activity **Define Partner Profiles**



STEP 4: Define Partner Profiles

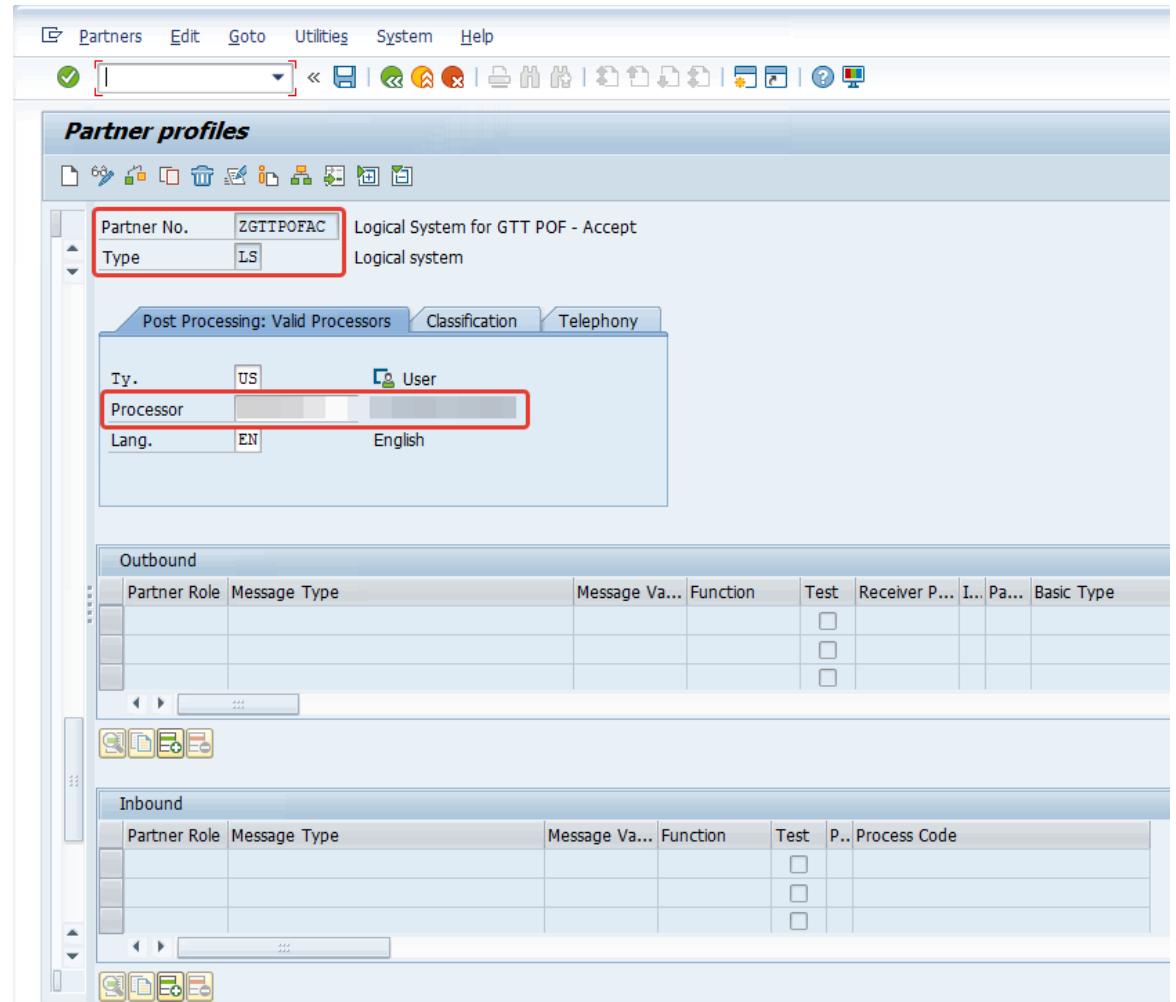
4-3: Choose **Partner Type LS** folder, and click **Create** to create a new partner profile



STEP 4: Define Partner Profiles

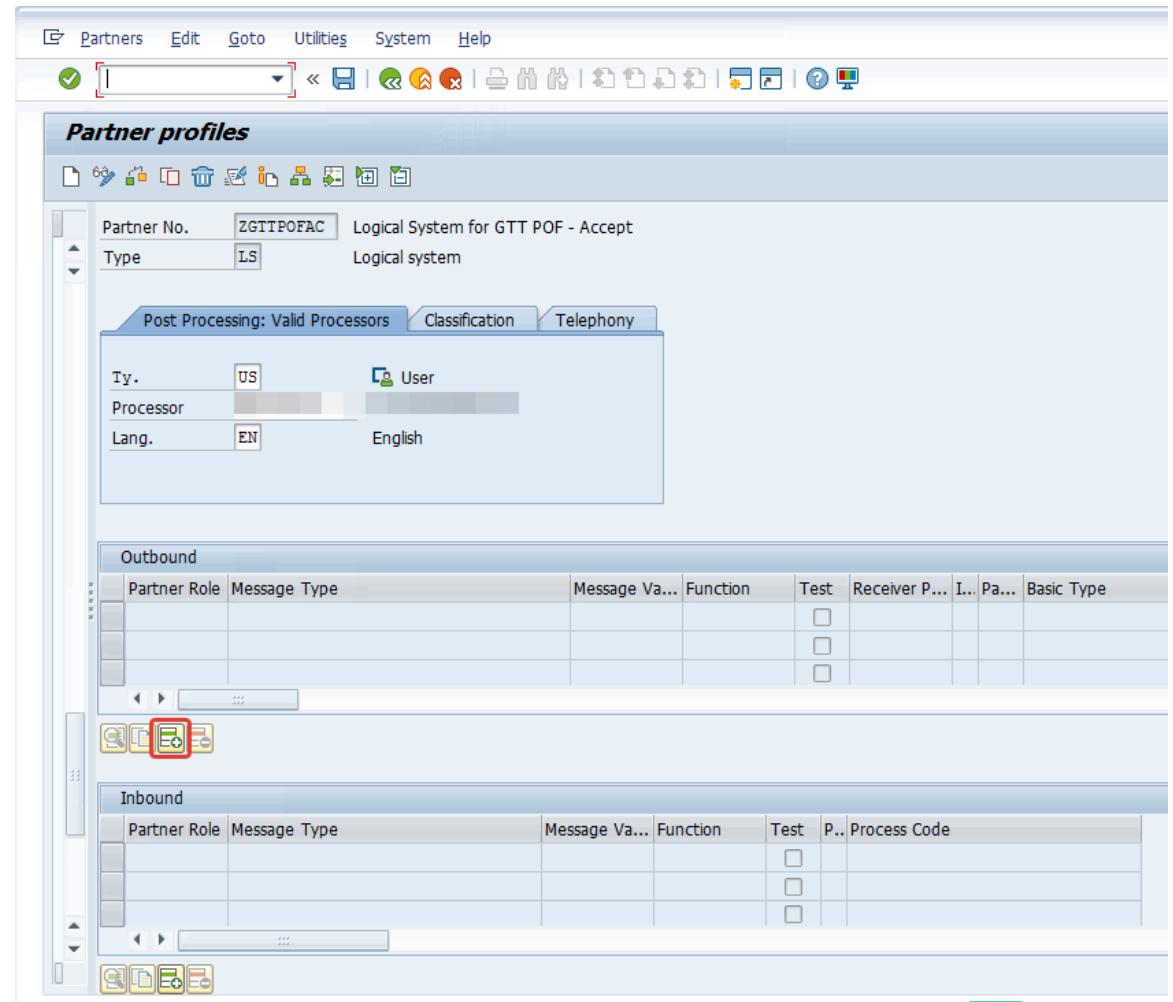
4-4: Fill in the **Partner No.** that you created in STEP 2

4-5: Fill in the **Processor** information



STEP 4: Define Partner Profiles

4-6: Click **Add** under **Outbound** box to create a new outbound parameter



STEP 4: Define Partner Profiles

4-7: Fill in the Message Type.

For the event:

Message Type: EVMSTA

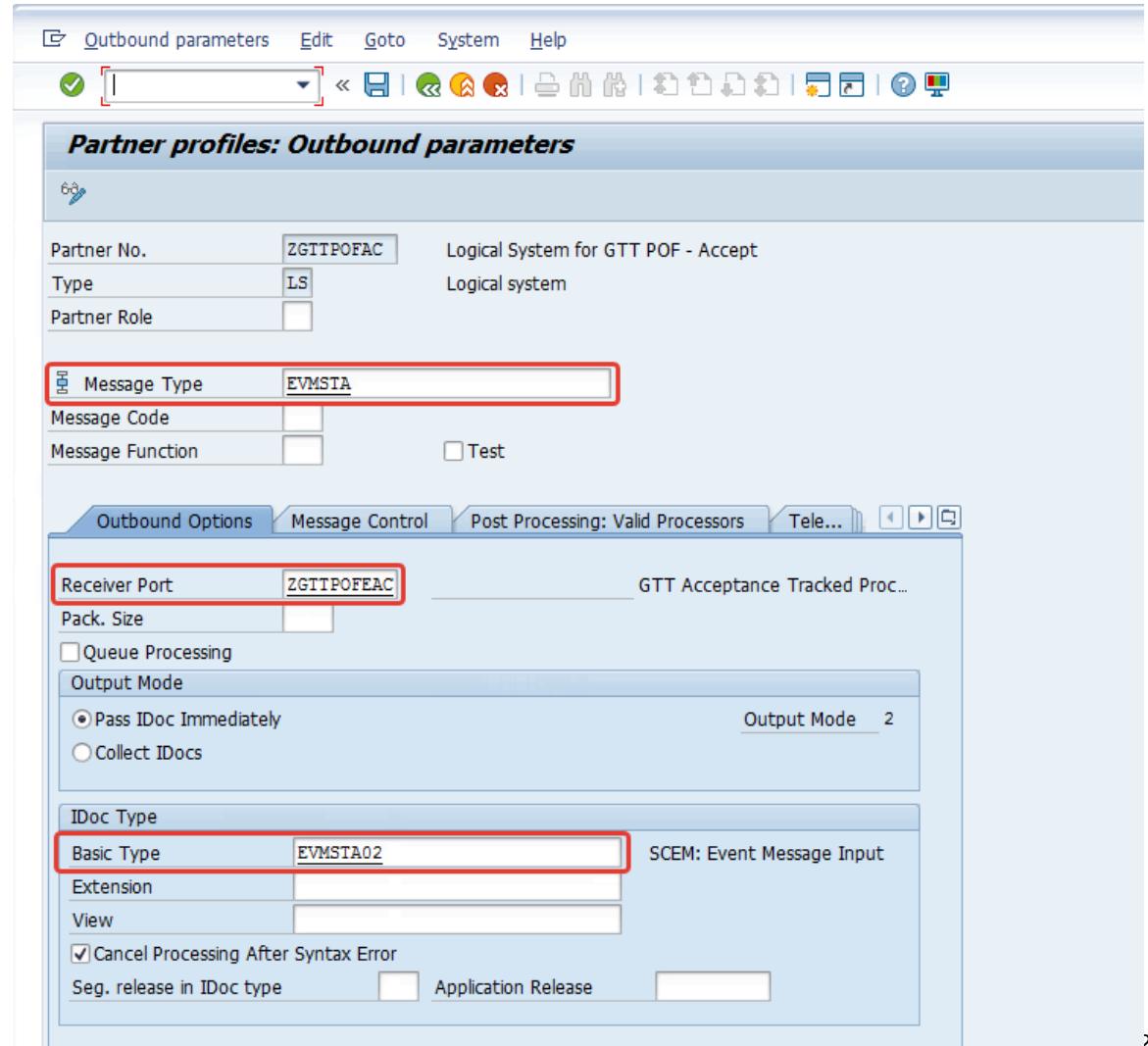
For the tracked Process:

Message Type: AOPOST

4-8: Fill in the Receiver Port, that you created in STEP 3

4-9: Save the configuration

Caution: In this step, you need to repeat steps 4-6 ~ 4-9 to add two outbound parameters, one for event and the other for tracked process.



B) Configuration and Implementation

- Basic

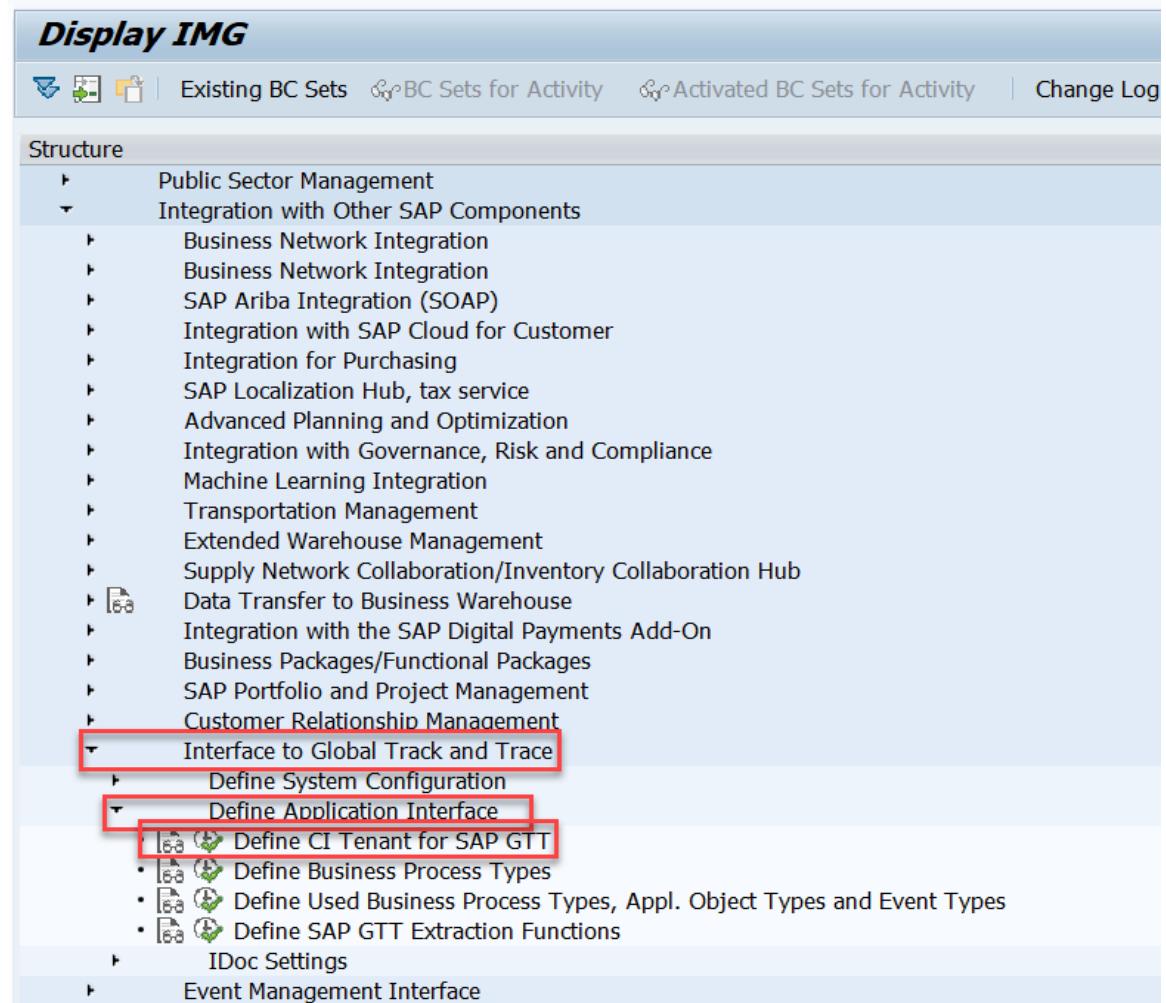
B2. Extractor Configuration



STEP 5: Define CI Tenant for GTT

5-1: In **Display IMG** page, click
Integration with Other SAP Components ->
Interface to Global Track and Trace ->
Define Application Interface

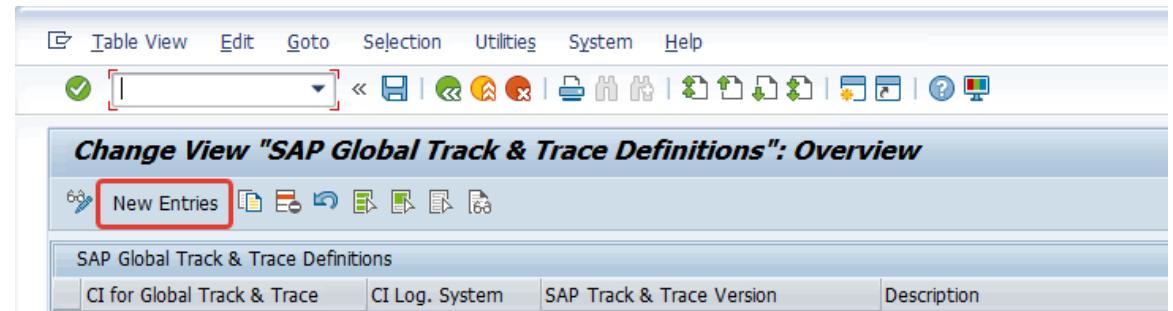
5-2: Choose activity
Define CI Tenant for SAP GTT



STEP 5: Define CI Tenant for GTT

5-3: Click **New Entries** to create a new CI tenant for GTT

5-4: Fill in the information for the new CI tenant. The **CI Log. System** is the logical system you created in STEP 2.



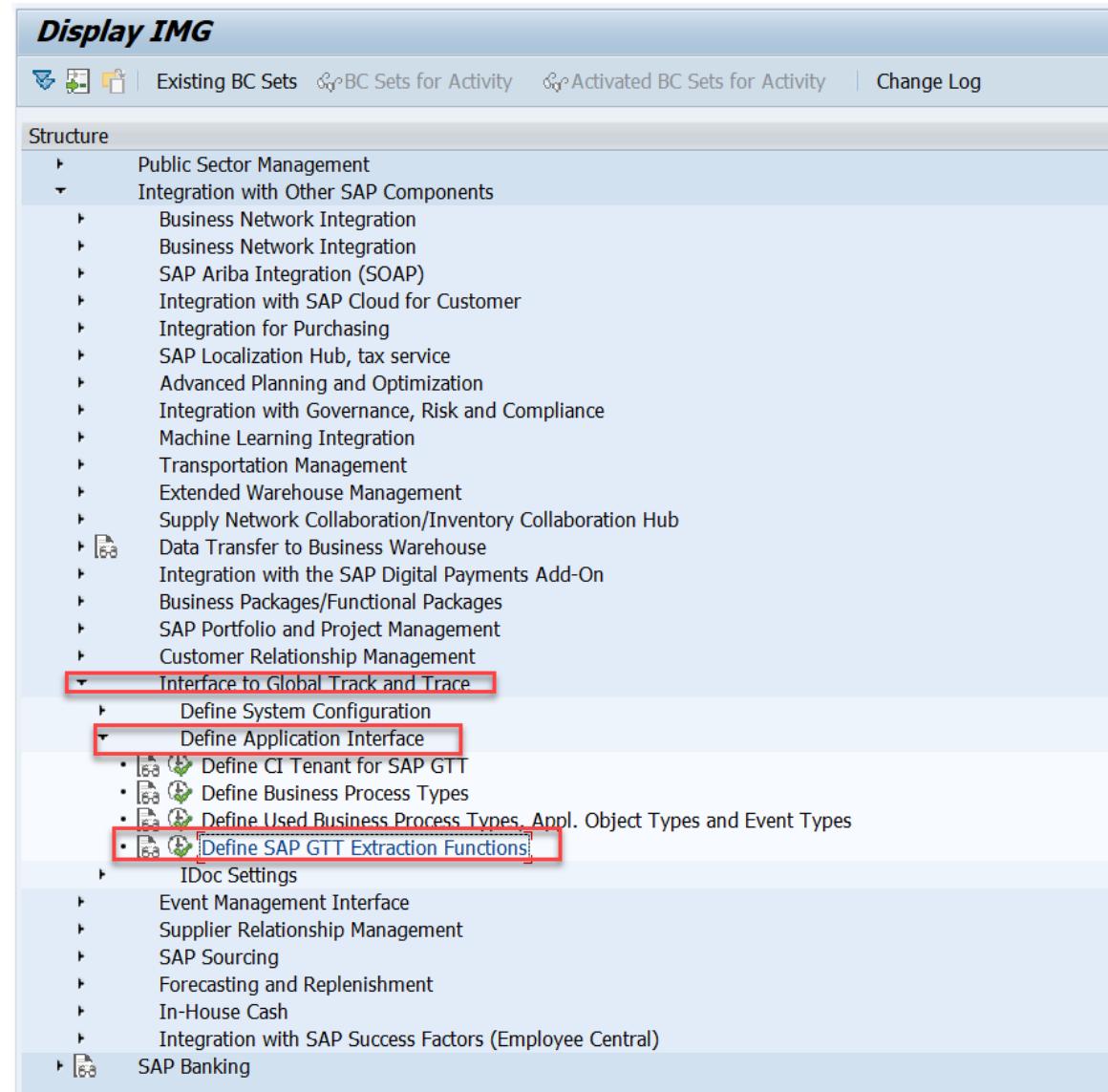
The screenshot shows the SAP Global Track & Trace Definitions overview screen. The top menu bar includes Table View, Edit, Goto, Selection, Utilities, System, and Help. Below the menu is a toolbar with various icons. A red box highlights the 'New Entries' button, which is located in the toolbar. The main title is 'Change View "SAP Global Track & Trace Definitions": Overview'. Below the title is a toolbar with icons for New Entries, Copy, Paste, etc. The main area displays a table titled 'SAP Global Track & Trace Definitions' with columns: CI for Global Track & Trace, CI Log. System, SAP Track & Trace Version, and Description. The 'Description' column for the first row contains the text 'CI For GTT Purchasing Order Sample APP - Acceptance'. The entire row for this entry is highlighted with a red box.

CI for Global Track & Trace	CI Log. System	SAP Track & Trace Version	Description
ZGTTPOFAC	ZGTTPOFAC	GTT1.0 Global Track & Trace	CI For GTT Purchasing Order Sample APP - Acceptance

STEP 6: Define GTT Extraction Functions

6-1: In **Display IMG** page, click
Integration with Other SAP Components ->
Interface to Global Track and Trace ->
Define Application Interface

6-2: Choose activity
Define SAP GTT Extraction Functions

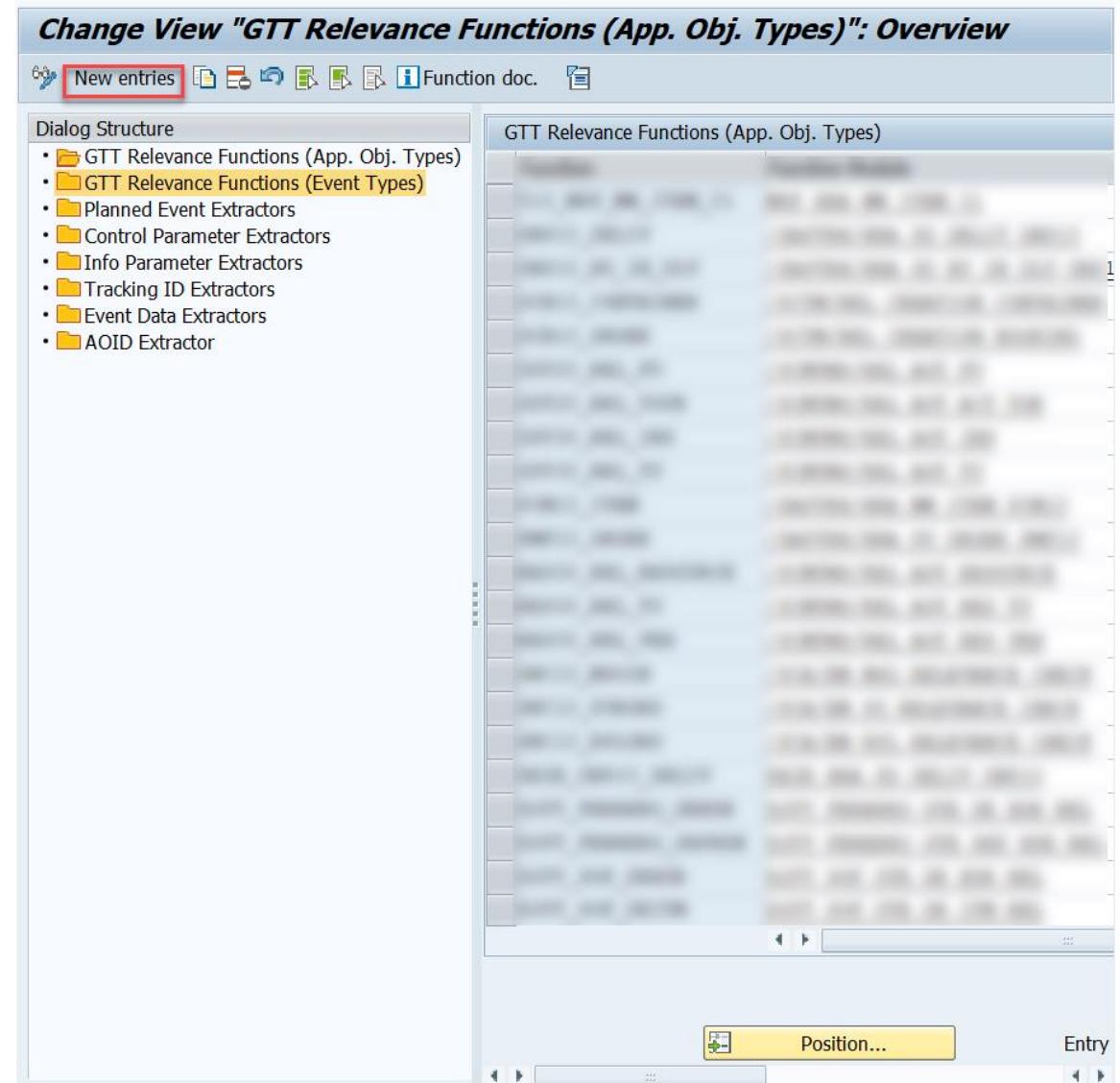


The screenshot shows the SAP Display IMG interface. The title bar reads "Display IMG". Below it is a toolbar with icons for search, refresh, and other functions, followed by links for "Existing BC Sets", "BC Sets for Activity", "Activated BC Sets for Activity", and "Change Log". The main area is titled "Structure" and contains a hierarchical list of SAP components. A red box highlights the path: "Integration with Other SAP Components" -> "Interface to Global Track and Trace" -> "Define Application Interface" -> "Define SAP GTT Extraction Functions". The final step, "Define SAP GTT Extraction Functions", is also highlighted with a red box.

- Public Sector Management
- Integration with Other SAP Components
 - Business Network Integration
 - Business Network Integration
 - SAP Ariba Integration (SOAP)
 - Integration with SAP Cloud for Customer
 - Integration for Purchasing
 - SAP Localization Hub, tax service
 - Advanced Planning and Optimization
 - Integration with Governance, Risk and Compliance
 - Machine Learning Integration
 - Transportation Management
 - Extended Warehouse Management
 - Supply Network Collaboration/Inventory Collaboration Hub
 - Data Transfer to Business Warehouse
 - Integration with the SAP Digital Payments Add-On
 - Business Packages/Functional Packages
 - SAP Portfolio and Project Management
 - Customer Relationship Management
- Interface to Global Track and Trace
 - Define System Configuration
 - Define Application Interface
 - Define CI Tenant for SAP GTT
 - Define Business Process Types
 - Define Used Business Process Types, Appl. Object Types and Event Types
 - Define SAP GTT Extraction Functions
 - IDoc Settings
 - Event Management Interface
 - Supplier Relationship Management
 - SAP Sourcing
 - Forecasting and Replenishment
 - In-House Cash
 - Integration with SAP Success Factors (Employee Central)
- SAP Banking

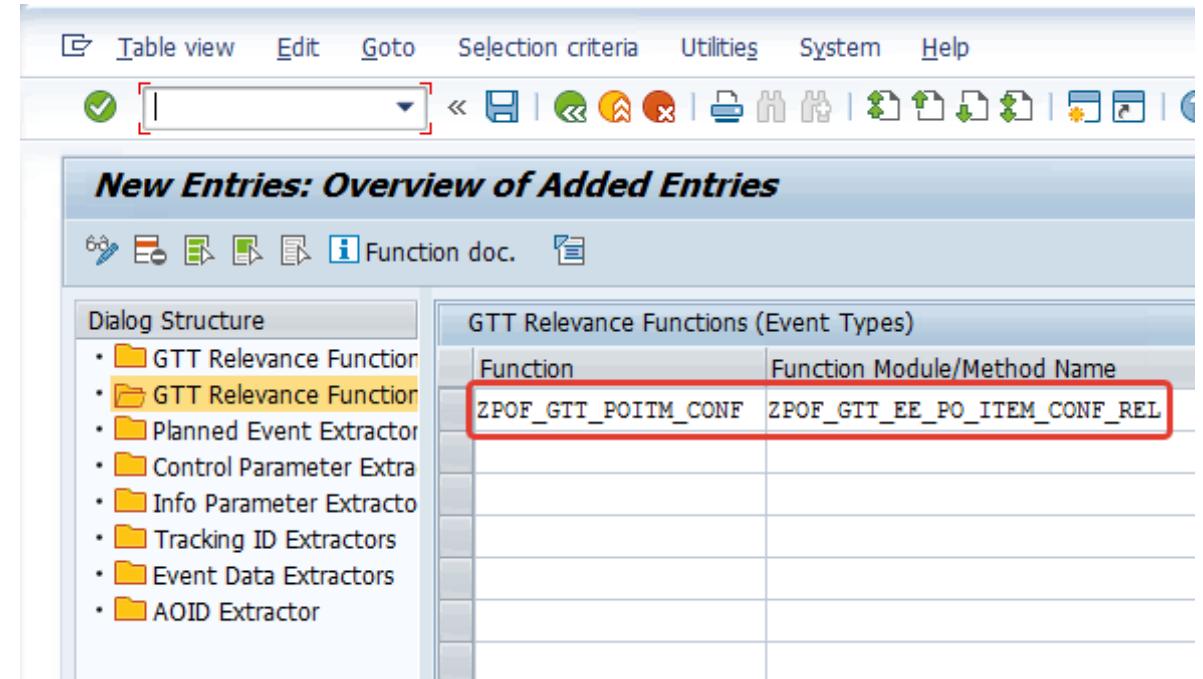
STEP 6: Define GTT Extraction Functions

6-3: Choose the type of Extraction Function you want to create from the **Dialog Structure**, and click **New entries**



STEP 6: Define GTT Extraction Functions

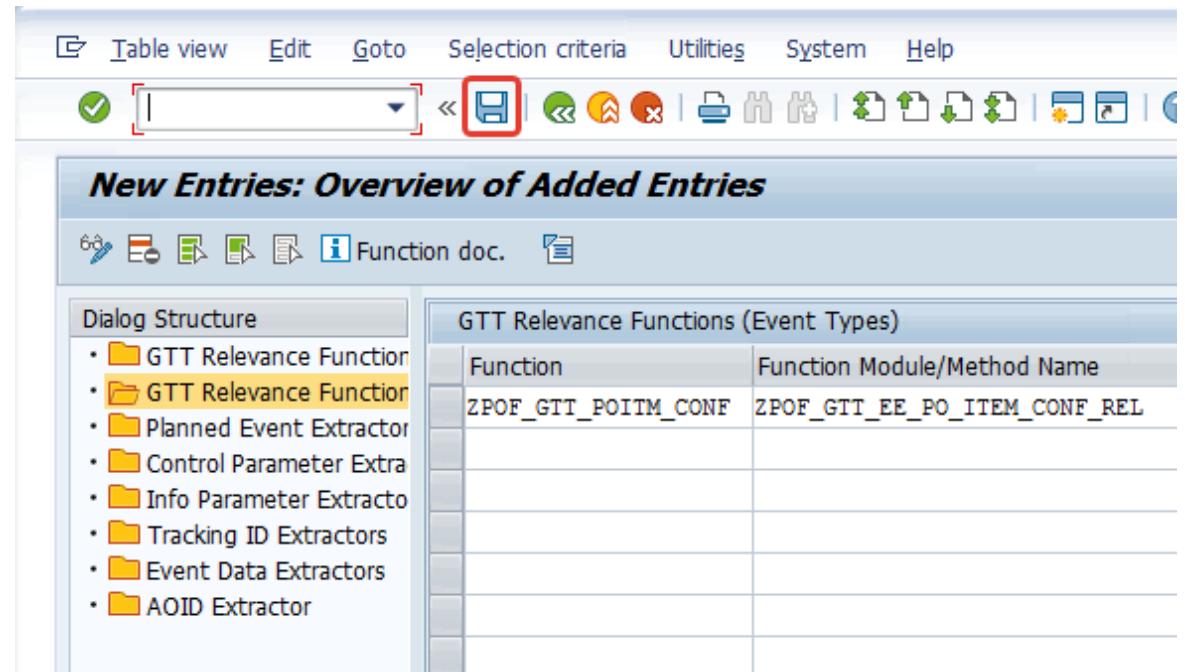
6-4: Input the **Function name** and **Function Module** for the newly created extraction function



GTT Relevance Functions (Event Types)	
Function	Function Module/Method Name
ZPOF_GTT_POITM_CONF	ZPOF_GTT_EE_PO_ITEM_CONF_REL

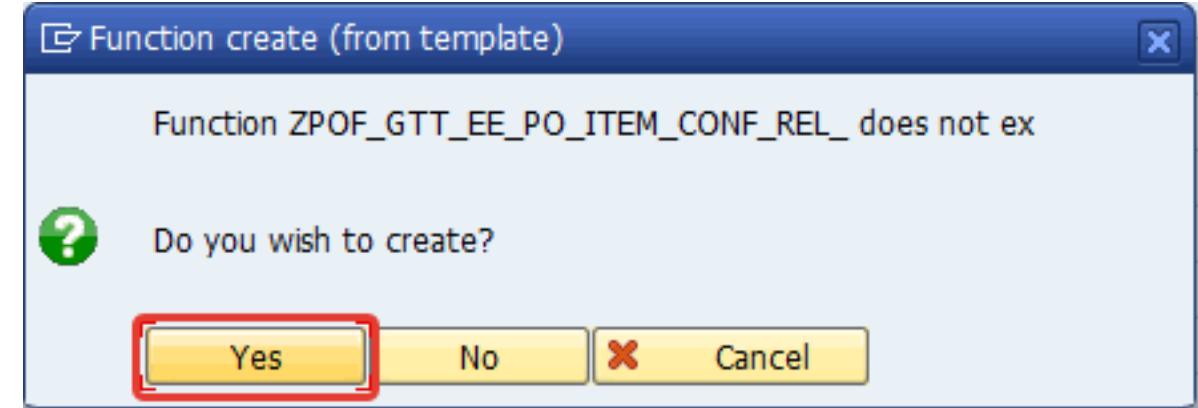
STEP 6: Define GTT Extraction Functions

6-5: Click **Save**



STEP 6: Define GTT Extraction Functions

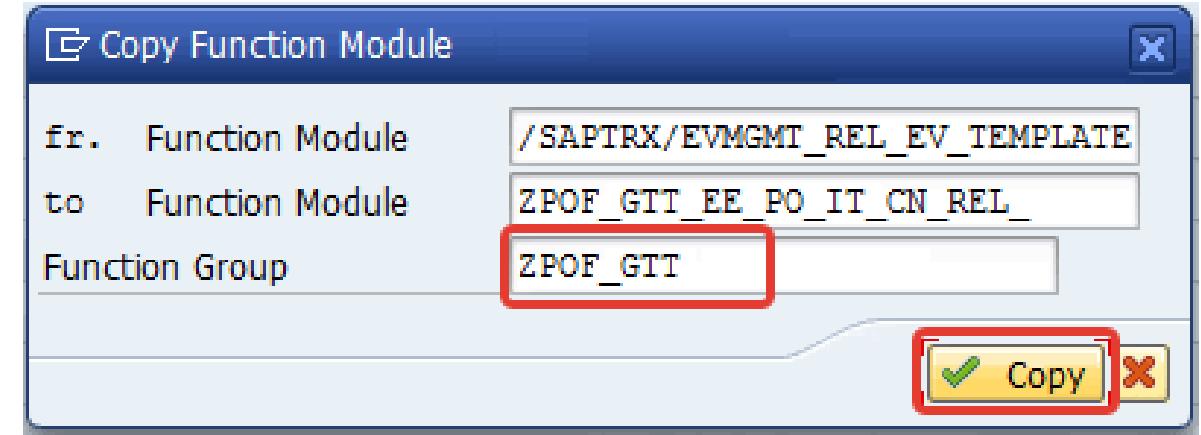
6-6: If the function module you use to create the extraction function has not been created yet, then a dialog reminds you to create the function module. Click **Yes** in the dialog box.



STEP 6: Define GTT Extraction Functions

6-7: Input the **Function Group** where the function module is to be created

6-8: Click **Copy**



STEP 6: Define GTT Extraction Functions

6-9: Use T-Code SE80 to check the function module you just created

Caution: More information on how to implement extraction functions and the relevant sample code is introduced later.

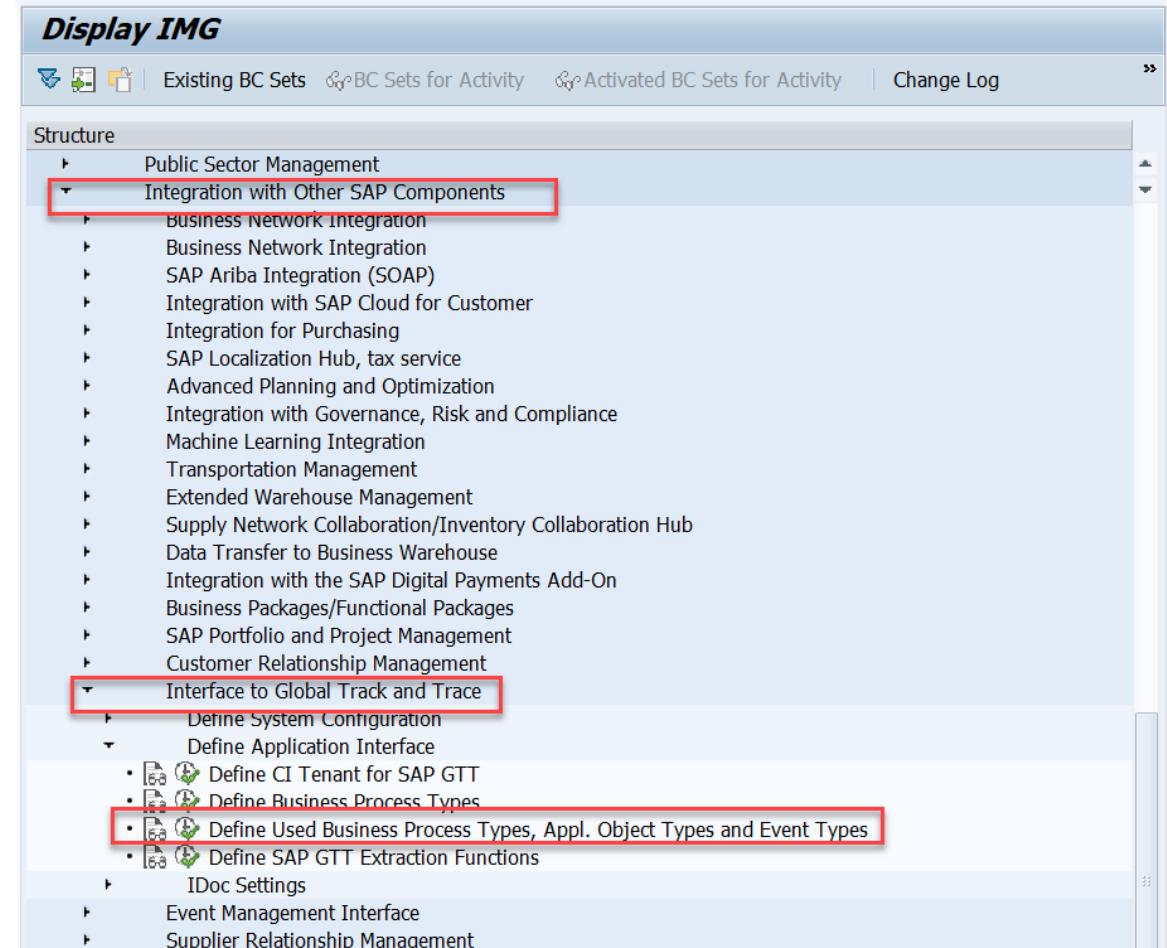
The screenshot shows the SAP SE80 Function Builder interface. The title bar reads "Function Builder: Display ZPOF_GTT_EE_PO_IT_CN_REL_". The left pane is a "Repository Browser" with a "Function Group" dropdown set to "ZPOF_GTT" (highlighted with a red box). The right pane displays the source code for the function module "ZPOF_GTT_EE_PO_IT_CN_REL_". The code is as follows:

```
1 FUNCTION ZPOF_GTT_EE_PO_IT_CN_REL_.  
2  
3 *--> Local Interface:  
4 IMPORTING  
5   REFERENCE(I_APPSYS) TYPE /SAPTRX/APPLSYSTEM  
6   REFERENCE(I_EVENT_TYPES) TYPE /SAPTRX/EVTYPES  
7   REFERENCE(I_ALL_APPL_TABLES) TYPE TRXAS_TABCONTAINER  
8   REFERENCE(I_EVENTTYPE_TAB) TYPE TRXAS_EVENTTYPE_TABS_WA  
9   REFERENCE(I_EVENT) TYPE TRXAS_EVT_CTAB_WA  
10 EXPORTING  
11   VALUE(E_RESULT) LIKE SY-BINPT  
12   TABLES  
13     C_LOGTABLE STRUCTURE BAPIRET2 OPTIONAL  
14 EXCEPTIONS  
15   PARAMETER_ERROR  
16   RELEVANCE_DETERM_ERROR  
17   STOP_PROCESSING  
18  
19 *-->  
20 * Top Include  
21 * TYPE-POOLS:trxas.  
22  
23  
24  
25  
ENDFUNCTION.
```

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-1: In Display IMG page, click
Integration with Other SAP Components ->
Interface to Global Track and Trace ->
Define Application Interface

7-2: Choose activity **Define Used Business Process Types, Appl. Object Types and Event Types**



STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

You can create event types and application object types for each business process type.

In the following:

- Steps 7-3 to 7-10 demonstrate how to create an *Event Type* for a given business process type
- Steps 7-11 to 7-21 demonstrate how to create an *Application Object Type* for a given business process type

Change View "Define Used Business Process Types": Overview		
New Entries		
Dialog Structure		
• Define Used Business Process Types	• Define Application Object Types	• Define Event Types
Bus. Proc. Type	Update Mode	BPT Process Mod
EPL_NOTIF	Update Task (▼ Active	
ESC_DELIV	Update Task ... ▼ Active	
ESC_FI_CLEARING	Update Task ... ▼ Active	
ESC_MATDOC	Update Task ... ▼ Active	
ESC_MM_INVOICE	Update Task ... ▼ Active	
ESC_PURORD	Update Task ... ▼ Active	
ESC_PURORD_FASHION	Update Task ... ▼ Active	
ESC_SHIPMT	Update Task ... ▼ Active	
ESC_SORDER	Update Task ... ▼ Active	
ESC_WRKORD	Update Task ... ▼ Active	
OCB10_ORDER	Dialog Update ▼ Active	
SNC_MSGIN	Dialog Update ▼ Active	
SNC_PURORD	Dialog Update ▼ Active	
SNC_RPLORD	Dialog Update ▼ Active	
TMS_INS	Update Task ... ▼ Active	
TMS_RES	Update Task ... ▼ Active	
TMS_TOR	Update Task ... ▼ Active	

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-3: Choose the business process type from the **Define Used Business Process Types** on the right side

7-4: Double click **Define Event Types**

Bus. Proc. Type	Update Mode	BPT Process Mode	Description
EPL_NOTIF	Update Task ...	Active	Notification in SAP R/3 Enterprise
ESC_DELIV	Update Task ...	Active	Delivery in SAP R/3 Enterprise
ESC_FI_CLEARING	Update Task ...	Active	FI Clearing in SAP R/3 Enterprise
ESC_MATDOC	Update Task ...	Active	Material Document in SAP R/3 Enterprise
ESC_MM_INVOICE	Update Task ...	Active	MM Invoice in SAP R/3 Enterprise
ESC_PURORD	Update Task ...	Active	Purchase Order in SAP R/3 Enterprise
ESC_PURORD_FASHION	Update Task ...	Active	Purchase Order (Seasonal Procurement) in SAP R/3 Enterprise 2.0
ESC_SHIPMT	Update Task ...	Active	Shipment (SAP R/3 Enterprise)
ESC_SORDER	Update Task ...	Active	Sales Order in SAP R/3 Enterprise
ESC_WRKORD	Update Task ...	Active	Workorder (Production, Service, Maintenance) in SAP R/3 Enterprise
OCB10_ORDER	Dialog Update	Active	Booking Order in Ocean Carrier Booking Process
SNC_MSGIN	Dialog Update	Active	SNC Inbound messages
SNC_PURORD	Dialog Update	Active	SNC Purchase Order
SNC_RPLORD	Dialog Update	Active	SNC Replenishment Order
TMS_INS	Update Task ...	Active	Instructions (SAP TM)
TMS_RES	Update Task ...	Active	Resources (SAP TM)
TMS_TOR	Update Task ...	Active	Transportation Order (SAP TM)

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-5: Click **New Entries** to create a new event type

The screenshot shows the SAP GUI interface for defining event types. The title bar reads "Change View 'Define Event Types': Overview". On the left, there is a "Dialog Structure" tree with nodes for "Define Used Business Pro", "Define Application Ot", and "Define Event Types". A red box highlights the "New Entries" button in the toolbar above the table. The main area displays a table titled "Define Event Types" with columns: Business Process Type, Event Type, and Description. The table contains three rows:

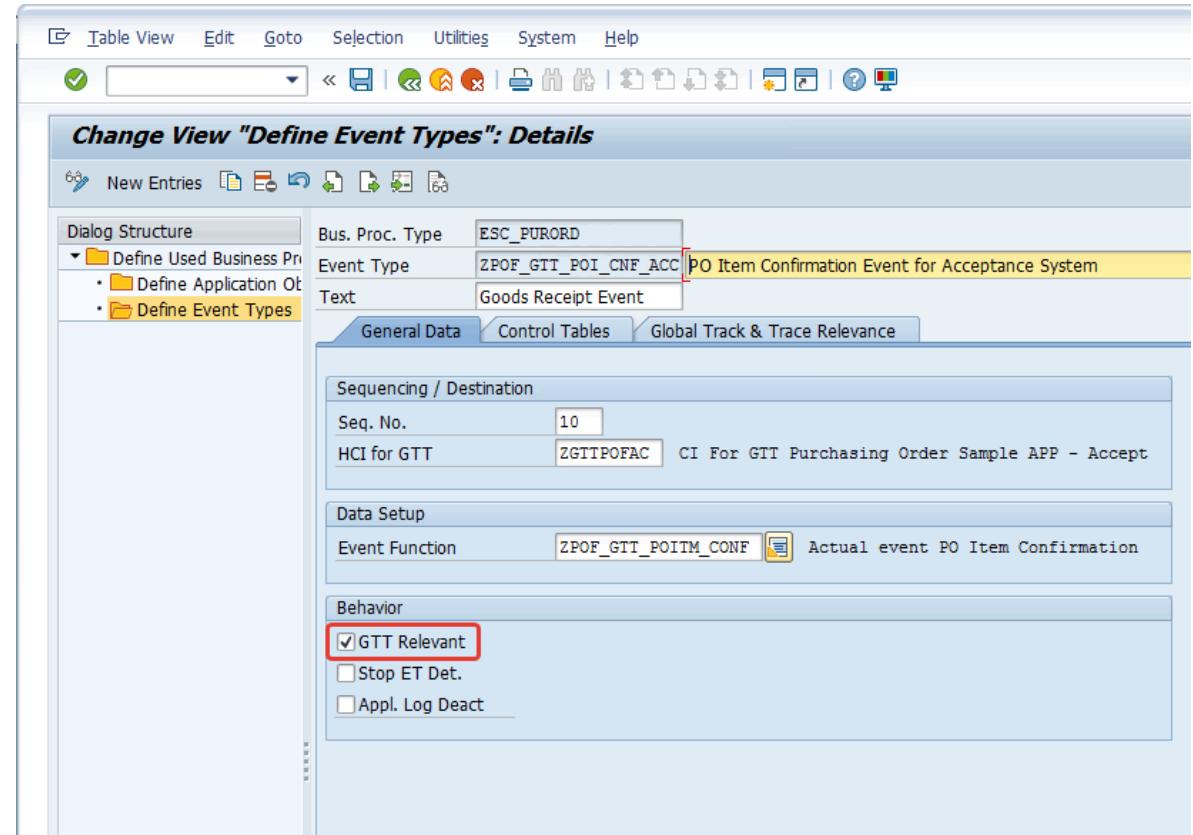
Business Process Type	Event Type	Description
ESC_PURORD	ZPOF_GTT_POI_DEL_ACC	PO Item Confirmation Event for Acceptance System
ESC_PURORD	ZPOF_GTT_POI_DEL_ACZ	PO Item Confirmation Event for Azure System
ESC_PURORD	ZPOF_GTT_POI_DEL_INT	PO Item Confirmation Event for Integration System

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-6: Fill in the **Event Type** and **Text** fields

7-7: Fill in the information required in the **General Data** tab. **HCI for GTT** is the CI Tenant you created in STEP 5. **Event Function** is the extractor function you created in STEP 6.

7-8: Check **GTT Relevant**



STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-9: Fill in the **Main Object Table** and **Master Table**.

Caution:

If the event type or application object type is on header level, then you only need to assign the **Main Object Table**. Otherwise, if the event type or application object type is on item level, then you need to assign the **Main Object Table** and **Master Table**, and assign the reference between the **Main Object Table** and **Master Table**.

Bus. Proc. Type	ESC_SHIPMT	
Event Type	ZPOF_GTT_SHH_ARR_ACC	Shipment Header Arrival Event for Acceptance System
Text	Arrival Event	
General Data Control Tables Global Track & Trace Relevance		
Data Source for Events		
Main Obj. Table	SHIPMENT_HEADER_NEW	Event on Header Level
Master Table		
Old Main Obj. Table	SHIPMENT_HEADER_OLD	
Old Master Table		

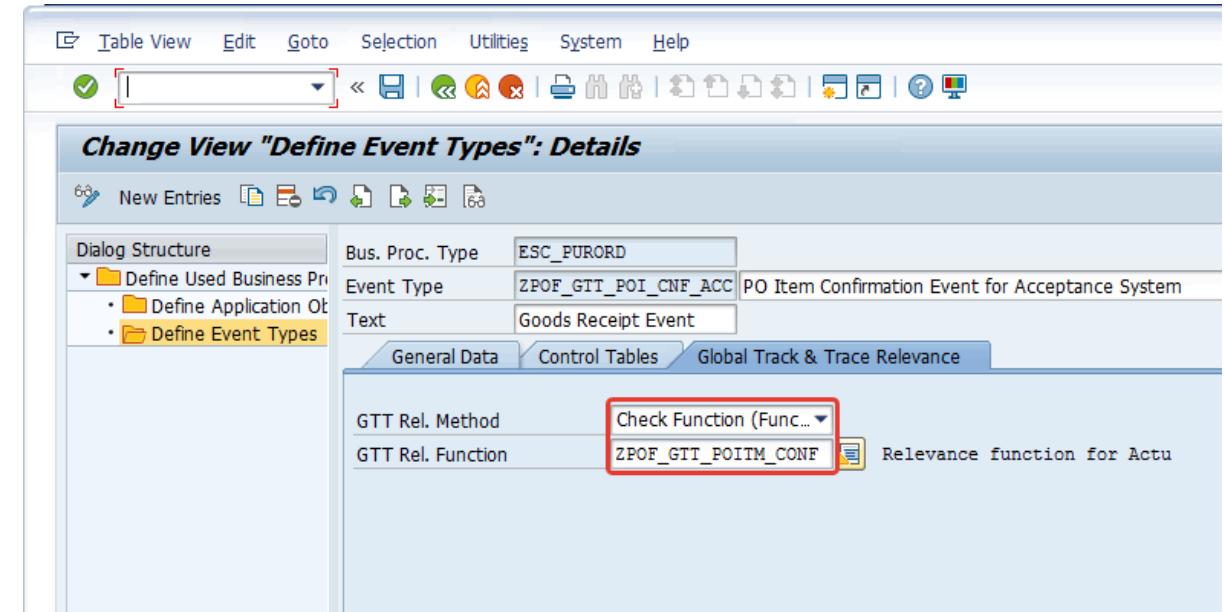
Bus. Proc. Type	ESC_PURORD	
Event Type	ZPOF_GTT_POT_CNF_ACC	PO Item Confirmation Event for Acceptance System
Text	Goods Receipt Event	
General Data Control Tables Global Track & Trace Relevance		
Data Source for Events		
Main Obj. Table	PURCHASE_ITEM_NEW	Event on Item Level
Master Table	PURCHASE_ORDER_HEADER_NEW	
Old Main Obj. Table	PURCHASE_ITEM_OLD	
Old Master Table	PURCHASE_ORDER_HEADER_OLD	
Reference Between Main and Master Table		
First Field Reference from Main to Master Table		
Uplink Field	EBELN	Uplink Mode <input checked="" type="checkbox"/> R
Uplink Target Fld	EBELN	Uplink Const

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-10: In the **Global Track & Trace Relevance** tab, choose the **GTT Relevance Method** you need.

If you choose the **GTT Relevance Method Check Function**, then you need to define a relevance function according to STEP 6, and fill in the relevance function name here.

Click **Save**.



STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

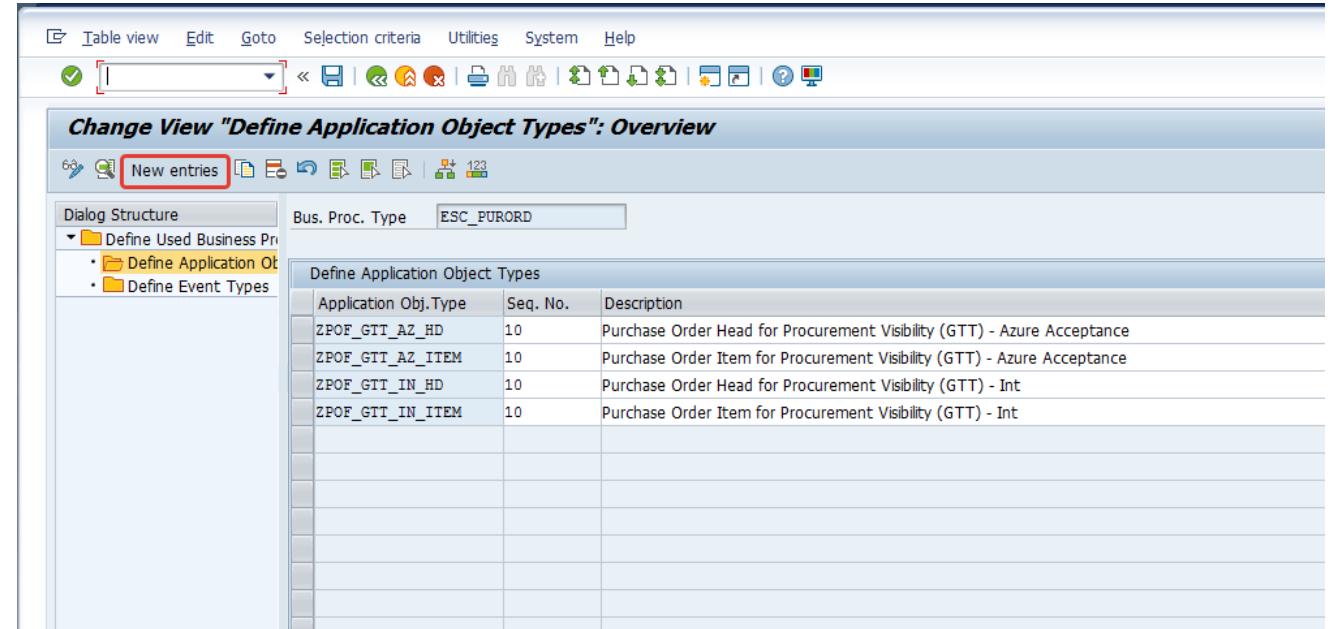
7-11: Choose the business process type from the **Define Used Business Process Types** on the right side

7-12: Double click **Define Application Object Types**

Bus. Proc. Type	Update Mode	BPT Process Mode	Description
EPL_NOTIF	Update Task...	Active	Notification in SAP R/3 Enterprise
ESC_DELIV	Update Task...	Active	Delivery in SAP R/3 Enterprise
ESC_FI_CLEARING	Update Task...	Active	FI Clearing in SAP R/3 Enterprise
ESC_MATDOC	Update Task...	Active	Material Document in SAP R/3 Enterprise
ESC_MM_INVOICE	Update Task...	Active	MM Invoice in SAP R/3 Enterprise
ESC_PURORD	Update Task...	Active	Purchase Order in SAP R/3 Enterprise
ESC_PURORD_FASHION	Update Task...	Active	Purchase Order (Seasonal Procurement) in SAP R/3 Enterprise 2.0
ESC_SHIPMT	Update Task...	Active	Shipment (SAP R/3 Enterprise)
ESC_SORDER	Update Task...	Active	Sales Order in SAP R/3 Enterprise
ESC_WRKORD	Update Task...	Active	Workorder (Production, Service, Maintenance) in SAP R/3 Enterprise
OCB10_ORDER	D Dialog Upd...	Active	Booking Order in Ocean Carrier Booking Process
SNC_MSGIN	D Dialog Upd...	Active	SNC Inbound messages
SNC_PURORD	D Dialog Upd...	Active	SNC Purchase Order
SNC_RPLORD	D Dialog Upd...	Active	SNC Replenishment Order
TMS_INS	Update Task...	Active	Instructions (SAP TM)
TMS_RES	Update Task...	Active	Resources (SAP TM)
TMS_TOR	Update Task...	Active	Transportation Order (SAP TM)

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-13: Click **New Entries** to create a new Application Object Type



The screenshot shows the SAP GUI interface for defining application object types. The title bar reads "Change View 'Define Application Object Types': Overview". The toolbar has various icons, and the "New entries" button is highlighted with a red box. The left sidebar shows a tree structure with "Define Used Business Pro" expanded, revealing "Define Application Obj" which is also highlighted with a red box. The main area displays a table titled "Define Application Object Types" with columns "Application Obj.Type", "Seq. No.", and "Description". The table contains four entries:

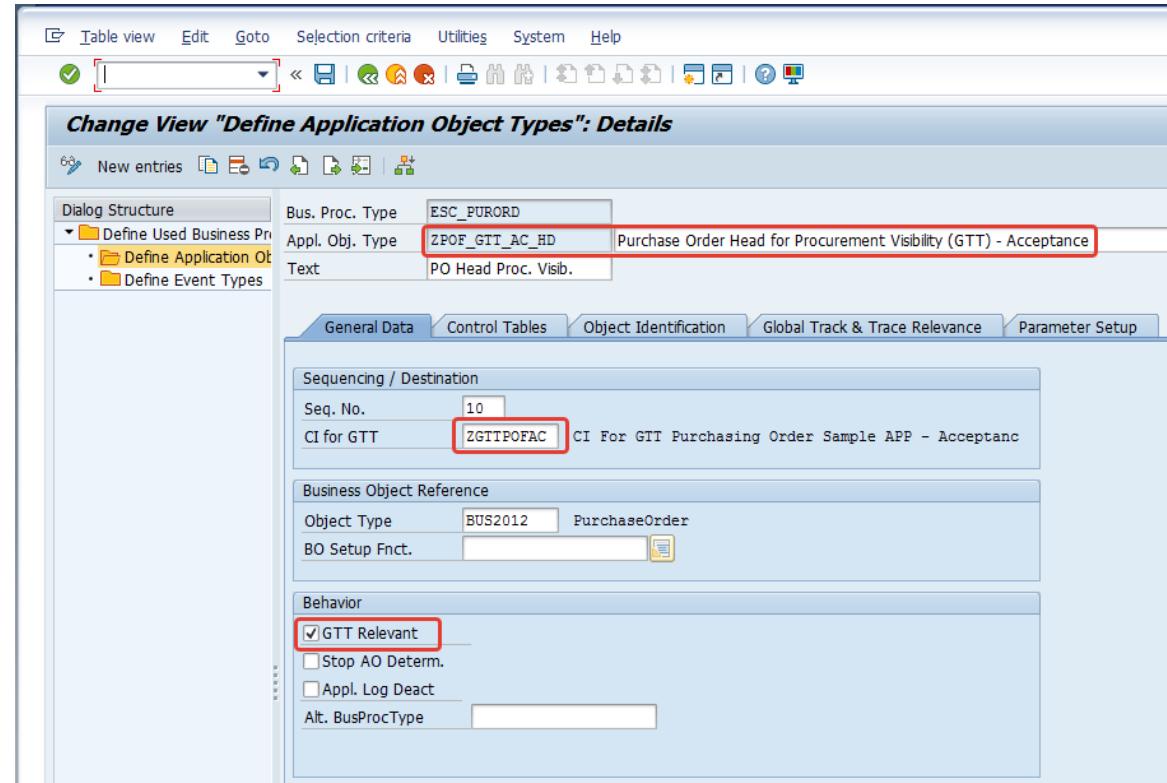
Application Obj.Type	Seq. No.	Description
ZPOF_GTT_AZ_HD	10	Purchase Order Head for Procurement Visibility (GTT) - Azure Acceptance
ZPOF_GTT_AZ_ITEM	10	Purchase Order Item for Procurement Visibility (GTT) - Azure Acceptance
ZPOF_GTT_IN_HD	10	Purchase Order Head for Procurement Visibility (GTT) - Int
ZPOF_GTT_IN_ITEM	10	Purchase Order Item for Procurement Visibility (GTT) - Int

STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-14: Fill in the Application Object Type and Text fields

7-15: Fill in the information required in the **General Data** tab. **CI for GTT** is the CI Tenant you created in STEP 5.

7-16: Check **GTT Relevant**

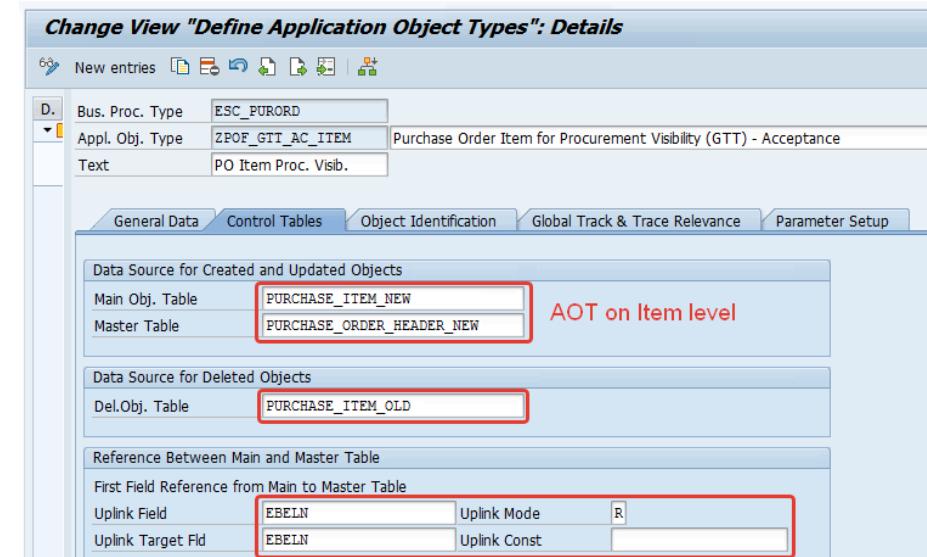
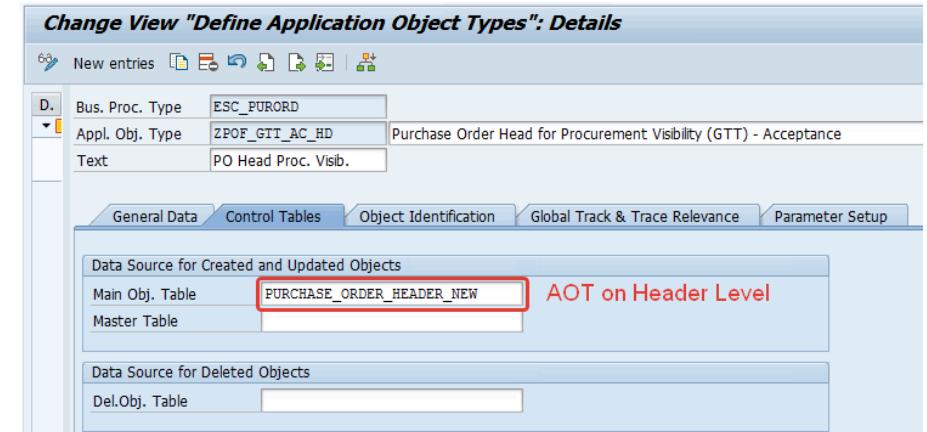


STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-17: Fill in the **Main Object table** and **Master Table**

Caution:

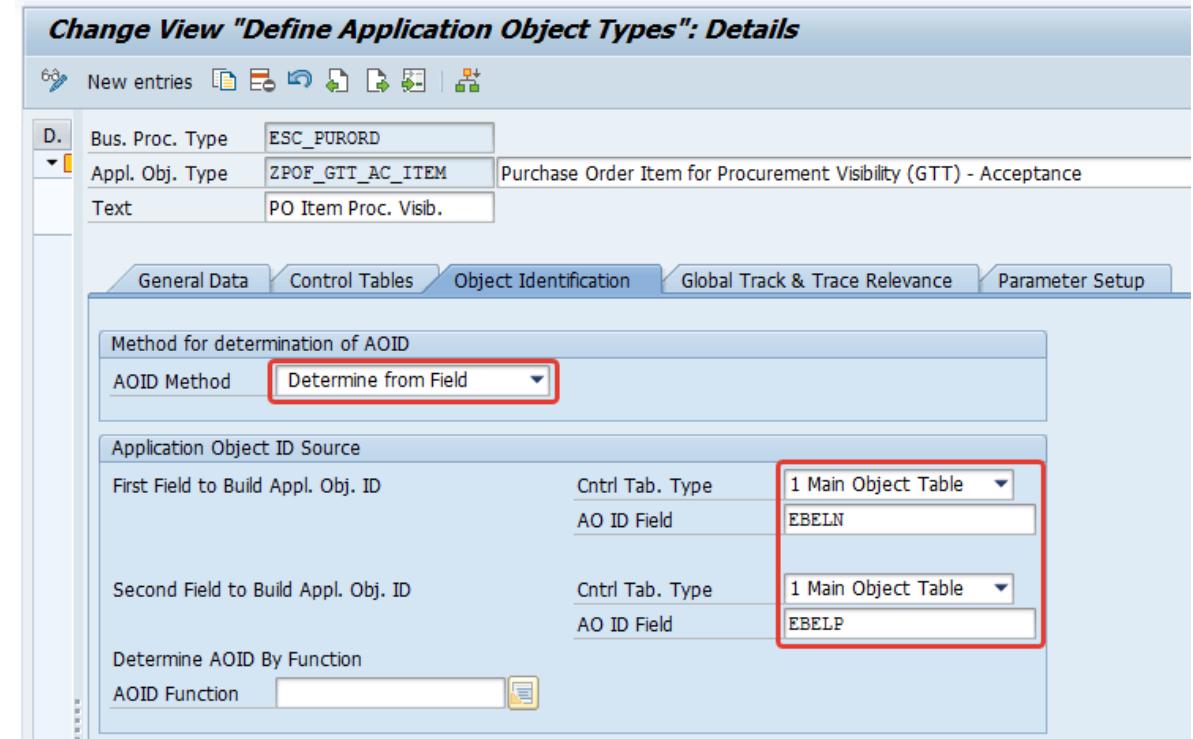
If the event type or application object type is on header level, then you only need to assign the **Main Object Table**. Otherwise, if the event type or application object type is on item level, then you need to assign the **Main Object Table** and **Master Table**, and assign the reference between the **Main Object Table** and **Master Table**.



STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-18: If there is no customized logic to determine the AOT ID, choose **Determine from Field**, use the key field to fill the AO ID fields

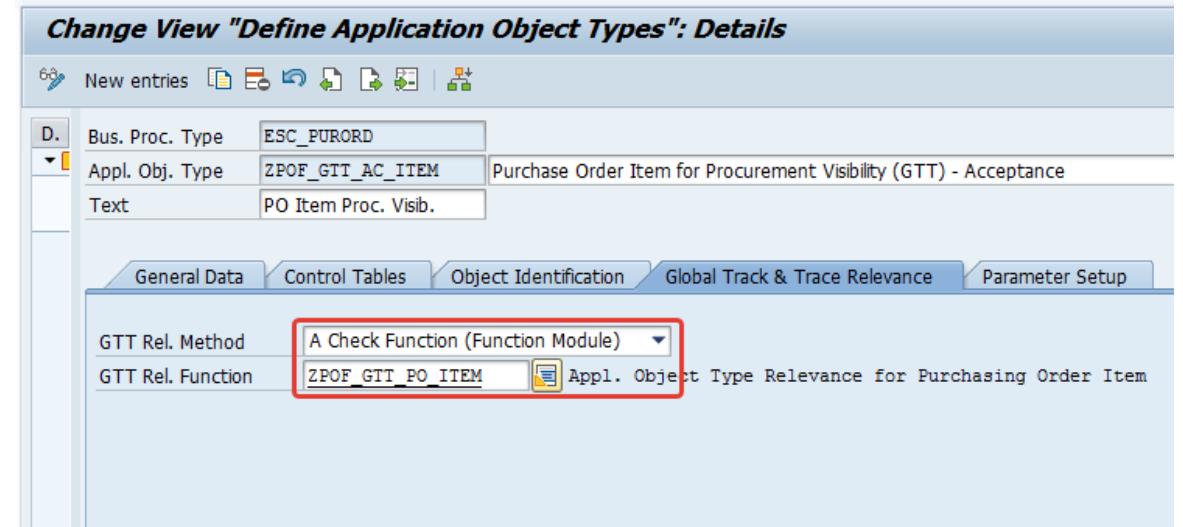
7-19: When choosing **Determine by Function**, you must enter the customized information in the AOID function field.



STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

7-20: In the **Global Track & Trace Relevance** tab, choose the **GTT Relevance Method** you need.

If you choose the **GTT Relevance Method Check Function**, then you need to define a relevance function according to STEP 6, and fill in the relevance function name here.



STEP 7: Define Used Business Process Types, Appl. Object Types and Event Types

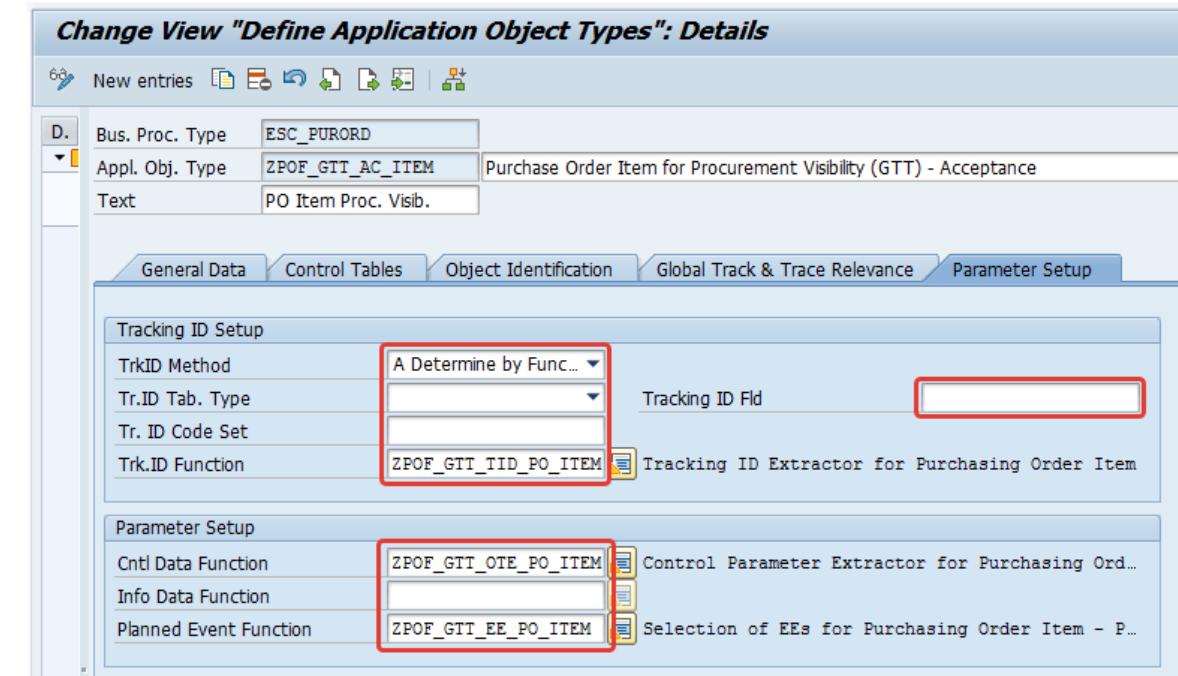
7-21: In the **Parameter Setup** tab, choose the **TrkID Method** as you need.

If you choose the **TrkID Method** as *Determine by Function*, then you need to define a tracking ID function according to STEP 6, and fill in the relevance function name here.

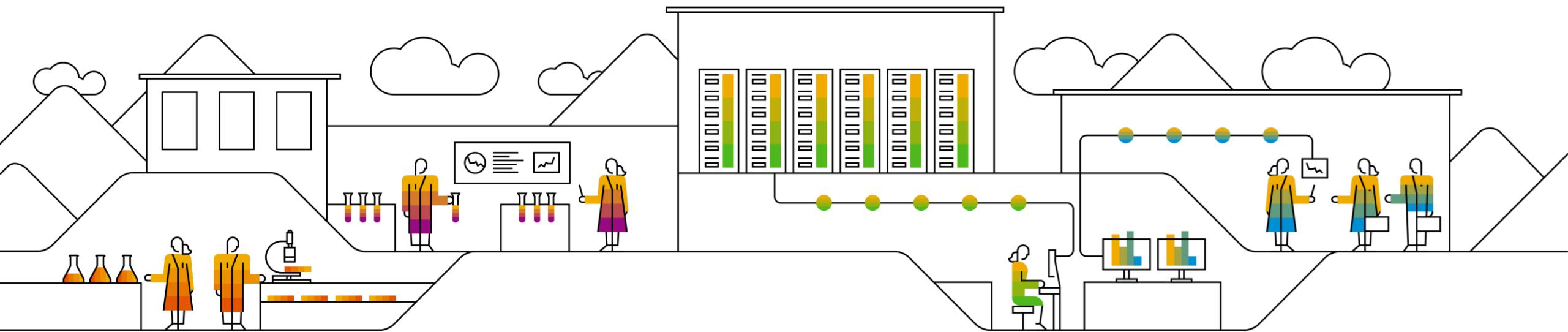
If no customized logic exists, for **TrkID Method** choose *Determine from Field*, then you need to fill the key field and name the Code Set for the AOT.

Fill in the extractor functions for **Control Data**, **Info Data(optional)**, **Planned Event**.

Click **Save**.



C) Download ABAP Code from GitHub



STEP 1: Install abapGit

You need to install abapGit before downloading codes from GitHub.

To install abapGit, follow the instructions on <https://docs.abapgit.org/guide-install.html>.

Make sure you **Install the standalone version** in your dev system.

When installation is complete, a new report is created, **ZABAPGIT_STANDALONE**.

The screenshot shows the abapGit documentation page with a navigation bar at the top. The main content area has two columns. The left column contains sections for Getting Started, Setup, Online Projects, Offline Projects, and Reference. The right column contains sections for Installation, Summary, Prerequisites, and a detailed 'Install standalone version' section. A red box highlights the 'Install standalone version' section, which contains step-by-step instructions and notes about the report name and usage.

abapGit documentation

Getting Started

- Installation
- Upgrading
- Uninstalling
- UI features

Setup

- SSL setup
- Proxy configuration
- Development version

Online Projects

- Installing online repo
- Keeping code up to date
- Uninstall repository
- First project
- Moving package into git
- Contributing to a project

Offline Projects

- Import zip
- Export zip

Reference

- Repo Settings (abapgit.xml)
- Supported object types
- Icon Legend
- User Exits
- Authorizations
- Namespaces

Installation

[Improve this page](#)

Summary #

abapGit exists in 2 flavours: *standalone* version or *developer* version.

- The *standalone* version is targeted at users. It consist of one (huge) program which contains all the needed code. You run the *standalone* version in transaction `SE38`, executing the program you created.
- The *developer* version is targeted at developers contributing to the abapGit codebase. It consists of all the ABAP programs/classes/interfaces/etc. of the abapGit project. You run the *developer* version with transaction `ZABAPGIT`.

Prerequisites #

abapGit requires SAP BASIS version 702 or higher.

Install standalone version #

1. Download the [ABAP code](#)(right click -> save-as) to a file.
2. Via `SE38` or `SE80`, create a new report named `ZABAPGIT_STANDALONE` (formerly `ZABAPGIT_FULL`). NB: Don't use the name `ZABAPGIT` if you plan to install the developer version.
3. In source code change mode, upload the code from the file using Utilities -> More Utilities -> Upload/Download -> Upload
4. Activate

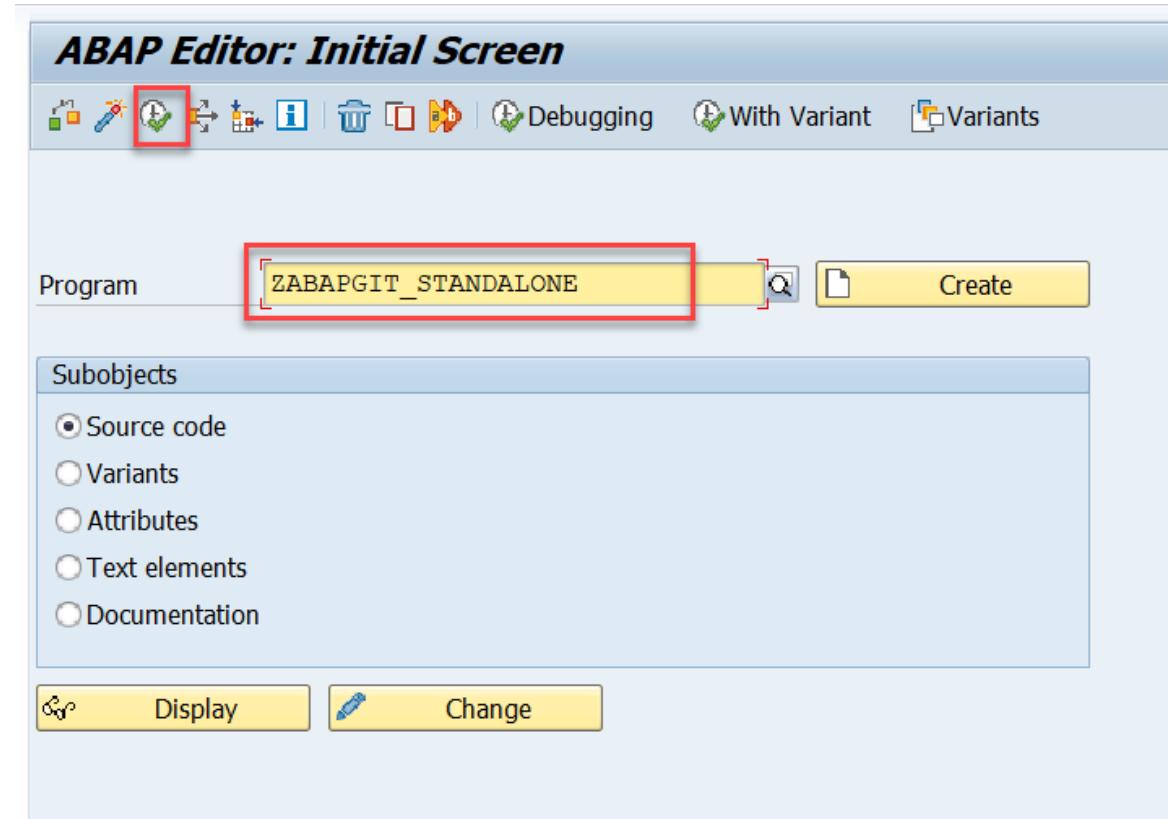
Typically, abapGit will only be used in the development system, so it can be installed in a local \$ package (e.g. `$ZABAPGIT`).

Now you can use abapGit by executing the report in transaction `SE38`.

STEP 2: Download ABAP Code

2-1: Enter T-code **SE38** and fill in the report name from STEP 1,
ZABAPGIT_STANDALONE

2-2: Click **Execute** to run the report



STEP 2: Download ABAP Code

2-3: Click **New Online** to download the code

The screenshot shows the ABAP GIT for GTT interface. At the top, there's a header bar with the title "ABAP GIT for GTT". Below it is a navigation bar with the "abapGit" logo and the text "Repository List". On the right side of the navigation bar are several buttons: "New Online" (highlighted with a red box), "New Offline", a delete icon, and a help icon. Below the navigation bar is a search bar labeled "Filter:" and two checkboxes: "Only Favorites" and "Detail". The main area is a table with columns: "Name", "Url", "Package", "Branch", and "Action". There are two rows of data in the table, both of which have their "Url" and "Package" fields blurred. The "Action" column for each row contains three small blue icons. At the bottom center of the page is the "abapGit" logo with the text "1.98.0". To the right of the logo is the text "js: OK".

STEP 2: Download ABAP Code

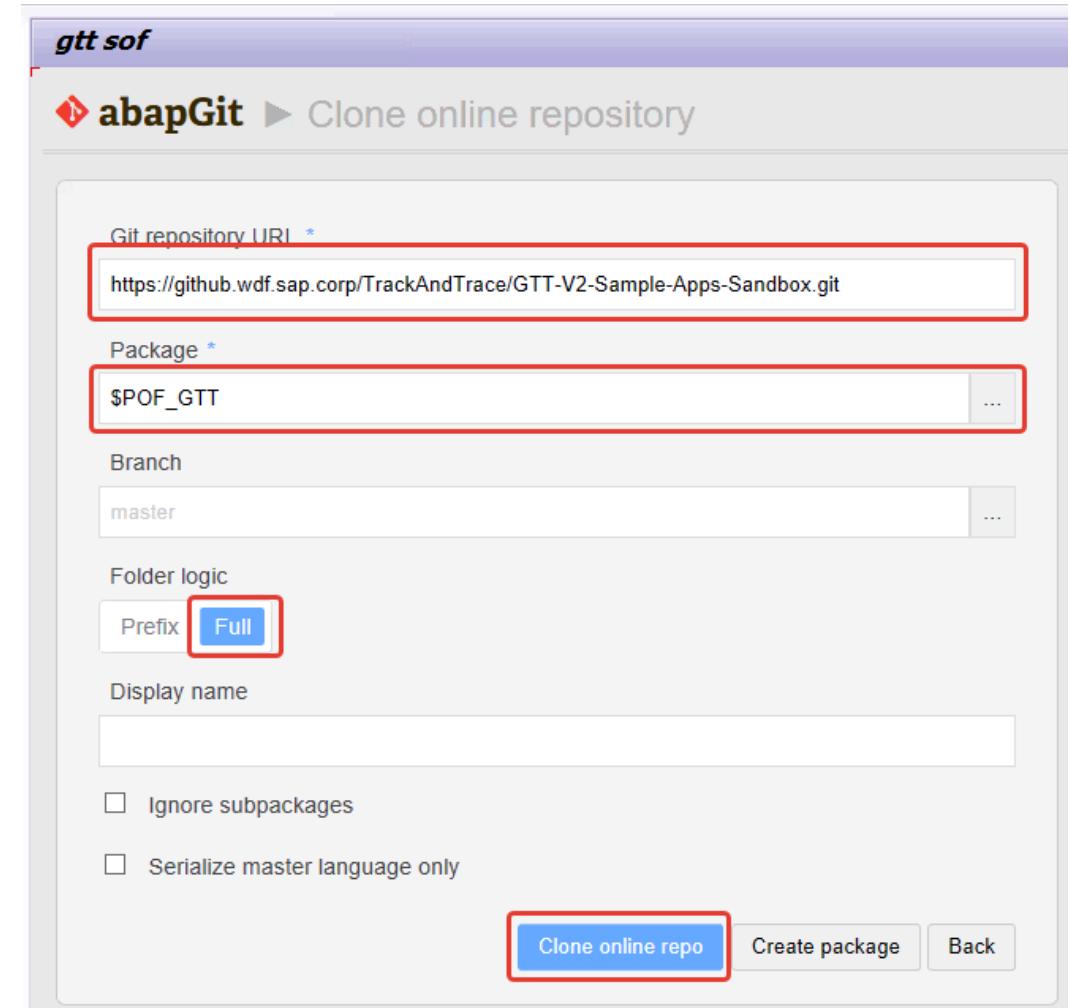
2-4: Fill in the **Git repository URL**:

<https://github.com/SAP-samples/logistics-business-network-gtt-samples.git>

2-5: Fill in the **Package** where you want to create the new ABAP code. If the package does not exist yet, click **Create package** to create it.

2-6: Set *Full* for **Folder Logic**

2-7: Click **Clone online repo** to download the code



STEP 2: Download ABAP Code

2-8: Click **Pull** to pull down the latest version code

TIP: Clicking **Pull** action will download the whole package of the sample codes. If the user want to download the specified folder's codes in Github, check details in Step 3-1 – 3-8

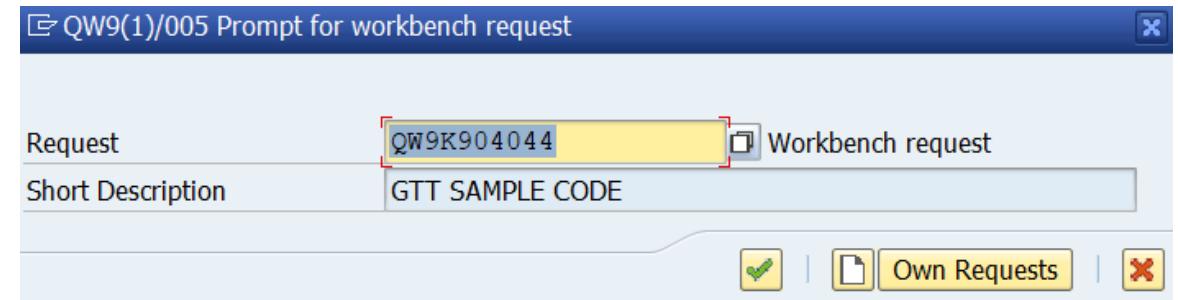
The screenshot shows the ABAP GIT for GTT interface. At the top, there is a navigation bar with the title "ABAP GIT for GTT" and a logo. Below it, a header bar displays the repository name "abapGit" and its URL "https://github.wdf.sap.corp/TrackAndTrace/GTT-V2-Sample-Apps-Sandbox.git". The repository has a commit hash "8f29562". On the right side of the header, there are buttons for "master" (highlighted in yellow), "\$POF_GTT", "Pull" (highlighted with a red box), "Stage", "Diff", "Branch", "Tag", "Advanced", "Refresh", and a gear icon.

The main area shows a table of files and their paths. The table has two columns: "File" and "Path". The "File" column contains icons for "DEV" (blue folder) and "CLAS" (green document). The "Path" column lists the full file paths. The last column contains "diff" buttons and status indicators (yellow, green, or grey).

File	Path	diff
DEVC \$POF_GTT	/lbn-gtt-pof-sample/ABAP/src/package.devco.xml	MM
CLAS ZCL_IM_POF_GTT_LE_SHIPMENT	/lbn-gtt-pof-sample/ABAP/src/zcl_im_pof_gtt_le_shipment.clas.abap	diff
	/lbn-gtt-pof-sample/ABAP/src/zcl_im_pof_gtt_le_shipment.clas.xml	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt00.abap	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt00.xml	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt01.abap	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt01.xml	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt10.abap	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt10.xml	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt11.abap	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt11.xml	diff
	/lbn-gtt-pof-sample/ABAP/src/zpof_gtt.fugr.lzpof_gtt20.abap	diff

STEP 2: Download ABAP Code

2-9: Assign the change to a change request. If you do not have any available change request, you need to create a new one.



STEP 2: Download ABAP Code

2-10: After you download the code, you can check them with T-code *SE80*.

The screenshot shows the SAP Repository Browser interface titled "Display Package". The "Package" dropdown menu is set to "\$POF_GTT", which is highlighted with a red box. The main area displays a hierarchical list of objects under the "\$POF_GTT" package, also enclosed in a red box. The objects listed include:

Object Name	Description
\$POF_GTT	ABAP sample package for global track and trace of LBN
Dictionary Objects	
Database Tables	ZPOF_GTT_EE_REL
Data Elements	ZPOF_KOSTA, ZPOF_PDSTK, ZPOF_PKSTA, ZPOF_WBSTA
Class Library	
Classes	ZCL_IM_POF_GTT_LE_SHIPMENT, ZPOF_GTT_LE_SHP_DLV_PROC_POI
Function Groups	ZPOF_GTT
Message Classes	ZPOF_GTT
Enhancements	
Classic BAdIs (Impl.)	ZPOF_GTT_LE_SHIPMENT
Enhancement Implementations	ZPOF_GTT_LE_SHP_DLV_PROC_POI

Step 3: Download ABAP Code Within the Specified Folder

3-1: If the user wants to download only the sample codes in the folder of 'Ibn-gtt-pof-sample' from Github instead of all downloaded, click **Fork** button to pop up a dialog window.

3-2: Click the user account and it will copy the newest version into the user's account.

The screenshot shows a GitHub repository page for 'SAP-samples / logistics-business-network-gtt-samples'. The 'Code' tab is selected. A red box highlights the 'Fork' button in the top right corner of the header, which has a value of 4. Below the header, there are buttons for Watch (9), Star (9), and Fork (4). The main area displays a list of commits:

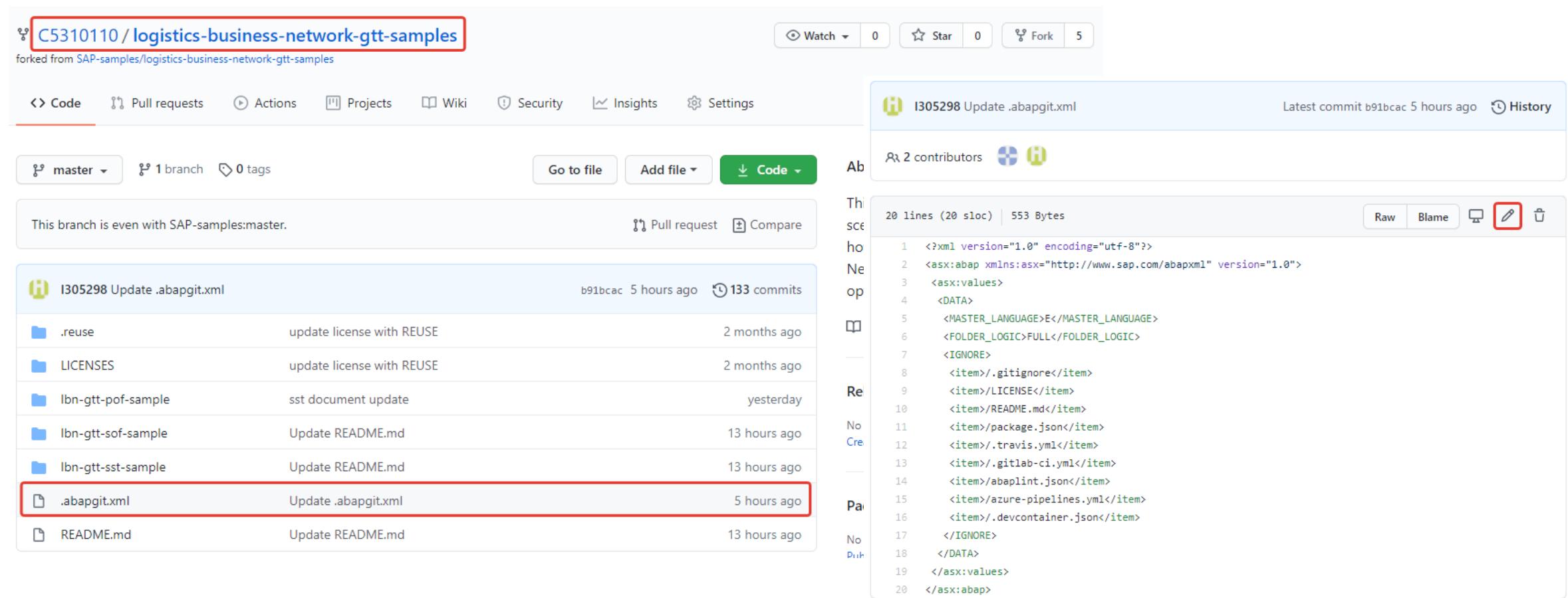
Commit	Message	Time Ago
I305298 Update .abapgit.xml	b91bcac 5 hours ago	133 commits
.reuse	update license with REUSE	2 months ago
LICENSES	update license with REUSE	2 months ago
Ibn-gtt-pof-sample	sst document update	yesterday
Ibn-gtt-sof-sample	Update README.md	13 hours ago
Ibn-gtt-sst-sample	Update README.md	12 hours ago
.abapgit.xml	Update .abapgit.xml	5 hours ago
README.md	Update README.md	12 hours ago

To the right, a modal dialog titled 'Fork logistics-business-network-gtt-samples' asks 'Where should we fork logistics-business-network-gtt-samples?'. It shows a list of accounts: 'I305298' (highlighted with a red box), 'Ibn', 'gtt', and 'sales-order-fulfillment'. Below the accounts are buttons for 'sample', 'sample-code', 'sap-logistics-business-network', 'global-track-and-trace', 'Ibn', 'gtt', and 'sales-order-fulfillment'.

Step 3: Download ABAP Code Within the Specified Folder

3-3: In the user account's repository, click the file '.abapgit.xml'

3-4: Click  button to edit the file



The screenshot shows a GitHub repository page for 'C5310110 / logistics-business-network-gtt-samples'. The repository has 0 stars and 5 forks. The main navigation bar includes Code, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation, it shows 'master' branch, 1 branch, 0 tags, Go to file, Add file, and a green 'Code' button. A message says 'This branch is even with SAP-samples:master.' with Pull request and Compare buttons. The commit list shows several commits, with the last one, 'I305298 Update .abapgit.xml', highlighted by a red box. This commit was made 5 hours ago by user 'b91bcac' and contains 133 commits. The commit details show changes to .reuse, LICENSES, lbn-gtt-pof-sample, lbn-gtt-sof-sample, lbn-gtt-sst-sample, .abapgit.xml, and README.md. The .abapgit.xml file content is displayed on the right, showing XML code for ABAP project structure. The code block has a red border around the entire content area.

```
<?xml version="1.0" encoding="utf-8"?>
<asx:abap xmlns:asx="http://www.sap.com/abapxml" version="1.0">
  <asx:values>
    <DATA>
      <MASTER_LANGUAGE>E</MASTER_LANGUAGE>
      <FOLDER_LOGIC>FULL</FOLDER_LOGIC>
      <IGNORE>
        <item>/.gitignore</item>
        <item>/.LICENSE</item>
        <item>/.README.md</item>
        <item>/.package.json</item>
        <item>/.travis.yml</item>
        <item>/.gitlab-ci.yml</item>
        <item>/.abaplint.json</item>
        <item>/.azure-pipelines.yml</item>
        <item>/.devcontainer.json</item>
      </IGNORE>
    </DATA>
  </asx:values>
</asx:abap>
```

Step 3: Download ABAP Code Within the Specified Folder

3-5: Add the sentence of '<STARTING_FOLDER>/lbn-gtt-pof-sample/ABAP/src/</STARTING_FOLDER>' like below

3-6: Commit change

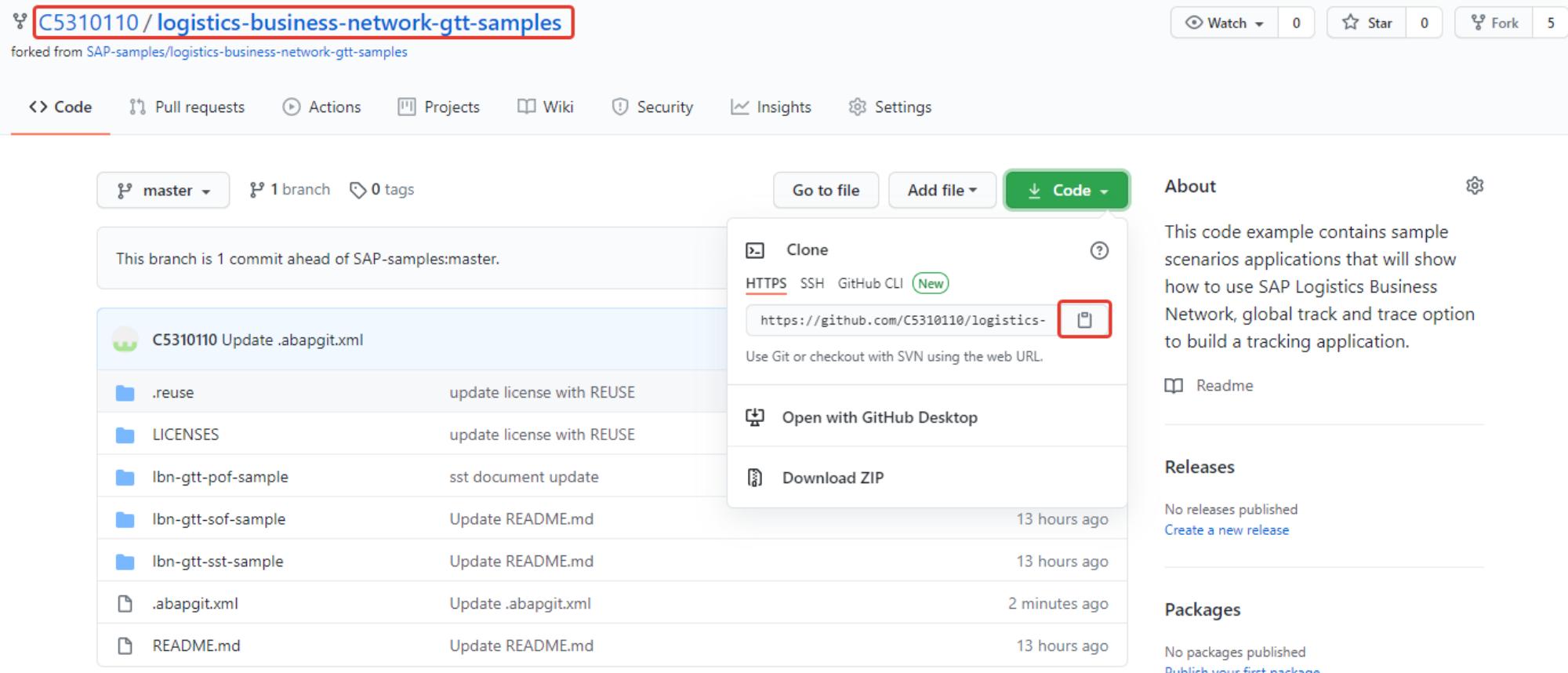
The screenshot shows a GitHub commit dialog for the file '.abapgit.xml'. The code editor on the left contains XML configuration for an ABAP git repository. A specific line, line 6, which contains the path '<STARTING_FOLDER>/lbn-gtt-pof-sample/ABAP/src/</STARTING_FOLDER>', is highlighted with a red box. The commit message field on the right is set to 'Update .abapgit.xml'. Below it, there's a text area for an optional extended description. At the bottom, two radio button options are shown: one selected ('Commit directly to the master branch') and one unselected ('Create a new branch for this commit and start a pull request'). A large green 'Commit changes' button is prominently displayed at the bottom of the dialog.

```
<?xml version="1.0" encoding="utf-8"?>
<asx:abap xmlns:asx="http://www.sap.com/abapxml" version="1.0">
<asx:values>
<DATA>
<MASTER_LANGUAGE>E</MASTER_LANGUAGE>
<STARTING_FOLDER>/lbn-gtt-pof-sample/ABAP/src/</STARTING_FOLDER>
<FOLDER_LOGIC>FULL</FOLDER_LOGIC>
<IGNORE>
<item>/.gitignore</item>
<item>/LICENSE</item>
<item>/README.md</item>
<item>/package.json</item>
<item>/.travis.yml</item>
<item>/.gitlab-ci.yml</item>
<item>/abaplint.json</item>
<item>/azure-pipelines.yml</item>
<item>/devcontainer.json</item>
</IGNORE>
</DATA>
</asx:values>
</asx:abap>
```

Step 3: Download ABAP Code Within the Specified Folder

3-7: Go to the root and copy the repository URL by clicking  button

3-8: Repeat Step 2-4 – 2.10



The screenshot shows a GitHub repository page for 'C5310110 / logistics-business-network-gtt-samples'. The URL 'https://github.com/C5310110/logistics...' is highlighted with a red box in the 'Code' dropdown menu.

Repository Overview:

- Code: master (selected), 1 branch, 0 tags
- This branch is 1 commit ahead of SAP-samples:master.
- Files listed:
 - C5310110 Update .abapgit.xml
 - .reuse update license with REUSE
 - LICENSES update license with REUSE
 - Ibn-gtt-pof-sample sst document update
 - Ibn-gtt-sof-sample Update README.md
 - Ibn-gtt-sst-sample Update README.md
 - .abapgit.xml Update .abapgit.xml
 - README.md Update README.md

Code Dropdown Menu:

- Clone (dropdown)
 - HTTPS (selected)
 - SSH
 - GitHub CLI (New)
- Use Git or checkout with SVN using the web URL.
<https://github.com/C5310110/logistics...> (highlighted with a red box)
- Open with GitHub Desktop
- Download ZIP

About:

This code example contains sample scenarios applications that will show how to use SAP Logistics Business Network, global track and trace option to build a tracking application.

Readme:

Releases:

No releases published
[Create a new release](#)

Packages:

No packages published
[Publish your first package](#)

Known Issue: Failure of New BADI Importing

Symptom: If the user wants to update an existing New BADI by report ZABAPGIT_STANDALONE, there is no error message, but BADI ZPOF_GTT_LE_SHP_DLV_PROC_POI is not updated or is not created if it was deleted before.

The screenshot displays two SAP interfaces: ABAPGit and the Repository Browser.

ABAPGit Interface (Left):

- Shows a repository named "POF GTT Offline".
- Contains files: ENHO (ZPOF_GTT_LE_SHP_DLV_PROC_POI), CLAS (ZCL_IM_POF_GTT_LE_SHIPMENT), and DEV (ZGTT_SAMPLE_POF).
- File details:
 - ENHO: /lbn-gtt-pof-sample/ABAP/src/zpof_gtt_le_shp_dlvc_proc_poi.enho.xml
 - CLAS: /lbn-gtt-pof-sample/ABAP/src/zcl_im_pof_gtt_le_shipment.clas.abap
 - DEV: /lbn-gtt-pof-sample/ABAP/src/package.devx.xml
- Import status: A successful import is shown for the ZGTT_SAMPLE_POF file.
- Message at the bottom: "The object will be created in the original language English (EN)".

Repository Browser Interface (Right):

- Shows a package named "ZGTT_SAMPLE_POF".
- Object tree:
 - ZGTT_SAMPLE_POF
 - Dictionary Objects
 - Class Library
 - Function Groups
 - Message Classes
 - Enhancements
 - Classic BAdIs (Impl.)
 - Enhancement Implementations
 - ZPOF_GTT_LE_SHP_DLV_PROC_POI
- A red box highlights the "ZPOF_GTT_LE_SHP_DLV_PROC_POI" folder in the "Enhancement Implementations" section.
- An information dialog box is open, stating: "Enhancement object cannot be read".

Known Issue: Failure of New BADI Importing

Solution: Set the breakpoint in Line 22 of class method: *CL_ENH_BADI_IMPL.Utility~ADD_BADIIMPLDIRENTRY* in transaction code *SE24*

Object Type: CL_ENH_BADI_IMPL.Utility

Properties

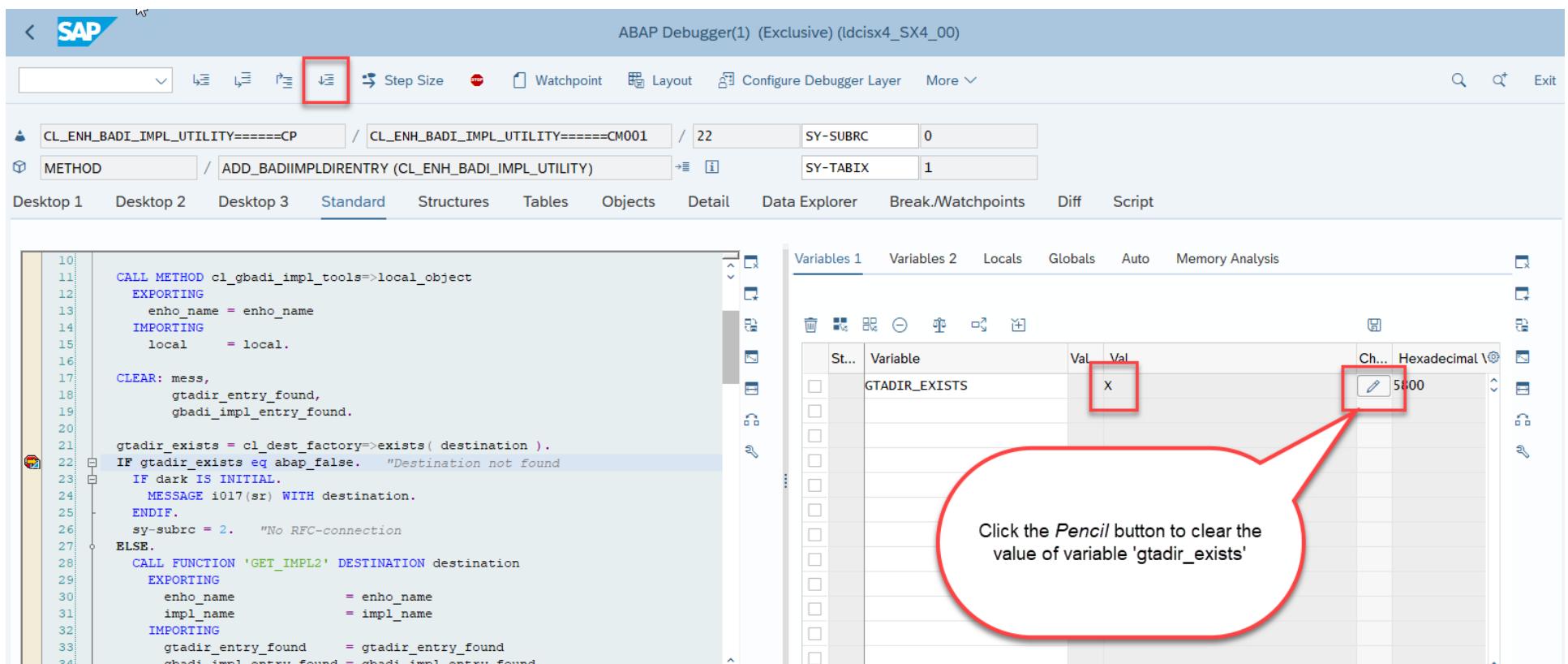
Method: ADD_BADIIMPLDIRENTRY

```
7: gbad_iimpl_entry_found TYPE char1,
8: gtadir_exists TYPE abap_bool,
9: destination(32) VALUE 'GTADIR_SERVER'.
10:
11: CALL METHOD cl_gbad_iimpl_tools=>local_object
12: EXPORTING
13: enho_name = enho_name
14: IMPORTING
15: local = local.
16:
17: CLEAR: mess,
18: gtadir_entry_found,
19: gbad_iimpl_entry_found.
20:
21: gtadir_exists = cl_dest_factory=>exists( destination ).
22: IF gtadir_exists eq abap_false. "Destination not found
23:   IF dard IS INITIAL.
24:     MESSAGE 1017(sr) WITH destination.
25:   ENDIF.
26:   sy-subrc = 2. "No RFC-connection
27: ELSE.
28:   CALL FUNCTION 'GET_IMPL2' DESTINATION destination
29:   EXPORTING
30:     enho_name = enho_name
31:     impl_name = impl_name
32:   IMPORTING
33:     gtadir_entry_found = gtadir_entry_found
34:     gbad_iimpl_entry_found = gbad_iimpl_entry_found
35:     gbad_iimpl_gtadir = gbad_iimpl_gtadir
```

Known Issue: Failure of New BADI Importing

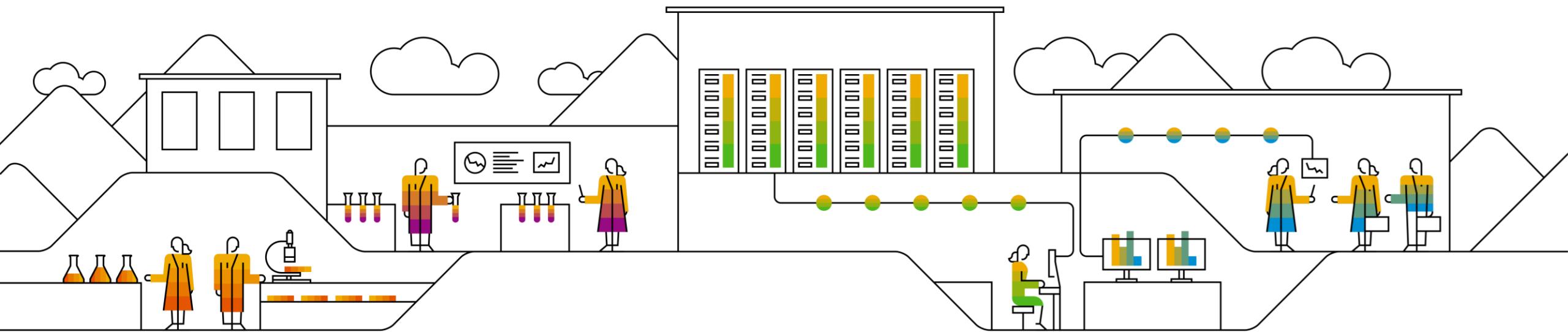
Solution: After setting the breakpoint, rerun the report ZABAPGIT_STANDALONE and repull the codes from the repository, the screen goes to Debug mode. Clear the variable 'gtadir_exists' and then execute by clicking button  , the BADI will be updated successfully.

TIP: debugging needs related developer authorization for which you should ask your system administrator



D) Configuration and Coding Guide

- Advanced



1: Maintain AOT type

When you are creating Application Object Type for one Business Process Type, make sure the AOT name must be the same as the name which is defined in the corresponding model in Manage Models app in GTT V2.

The image displays two SAP application screenshots side-by-side. On the left is the 'Change View "Define Application Object Types": Details' screen from the SAP AOT. It shows a table with columns: Bus. Proc. Type (ESC_PURORD) and Appl. Obj. Type (ZPOF_GTT_AC HD). The 'Appl. Obj. Type' field is highlighted with a red box. Below the table are tabs for General Data, Control Tables, Object Identification, Global Track & Trace Relevance, and Parameter Setup. Under 'General Data', there are sections for Sequencing / Destination (Seq. No. 10, CI for GTT ZGTTPOFAC) and Business Object Reference (Object Type BUS2012 PurchaseOrder, BO Setup Fnct.). On the right is the 'Model Details' screen from the Manage Models app. It shows a navigation bar with SAP logo, Model Details (Active), Internal - Test, and tabs for Tracked Process, Field Type Pool, Event Type Pool, Code List, IDOC Integration (highlighted with a red box), Visibility Provider Integration, Planned Event Extension, and Event to Action. The 'Tracked Process' section shows 'PurchaseOrder' selected. The 'IDOC Integration' section shows 'Tracked Process Mapping' with 'ERP Object Type: Others' and 'Application Object Type: ZPOF_GTT_AC HD' (highlighted with a red box). The 'Tracked Process / Events (1)' table lists 'Tracked Process' as PurchaseOrderEvent and 'Event Code' as E1EHPAO. The 'Fields' table maps fields like purchaseOrderNo, supplierId, plannedDeliveryDate, netValue, currency, incotermsVersion, and incoterms to IDOC segments like E1EHPCP and YN_PO_NUMBER, YN_PO_SUPPLIER_ID, YN_PO_DELIVERY_DATE, YN_PO_NET_VALUE, YN_PO_CURRENCY, YN_PO_INCOTERMS_VERSION, and YN_PO_INCOTERMS.

2: Maintain Tracking ID Type

In the AOT you maintained, make sure the Tracking ID Type is the same as the name which is defined in the corresponding process type of the model in Manage Models app in GTT V2.

If the Tracking ID Type is determined by Field, then input the value source field in the Tracking ID field, and the Code Set which is referring to the Tracking ID Type for the AOT like below.

Change View "Define Application Object Types": Details

D. Bus. Proc. Type: ESC_PURORD
Appl. Obj. Type: ZPOF_GTT_AC_HD [Purchase Order Head for Procurement Visibility (GTT) - Acceptance]
Text: PO Head Proc. Visib.

General Data Control Tables Object Identification Global Track & Trace Relevance Parameter Setup

Tracking ID Setup
TrkID Method: B Determine from Field
Tr.ID Tab. Type: 1 Main Object Table
Tr. ID Code Set: PURCHASE_ORDER
Trk.ID Function:

Parameter Setup
Cntl Data Function: ZPOF_GTT_OTE_PO_HD
Info Data Function:
Planned Event Function: ZPOF_GTT_EE_PO_HD

SAP Model Details Internal - Test

pof Active Purchase Order Fulfillment Namespace: com.lbngttsamples.gtt.app.pof Correlation Level: 4

Tracked Process Field Type Pool Event Type Pool Code List IDOC Integration Visibility Provider Integration Planned Event Extension Event to Action

Items (6) Create Edit Delete

PurchaseOrder
Description: Purchase Order
Tracking Id Type: PURCHASE_ORDER

PurchaseOrderItem
Description: PurchaseOrderItem
Tracking Id Type: PURCHASE_ORDER_ITEM

InboundDelivery
Description: Inbound Delivery
Tracking Id Type: INBOUND_DELIVERY

InboundDeliveryItem
Description: Inbound Delivery Item
Tracking Id Type: INBOUND_DELIVERY_IT

Create Tracked Process

Name: * PurchaseOrder
Description: Purchase Order
Tracking Id Type: PURCHASE_ORDER

OK Cancel

3: Make the Customization Logic in the Function Modules and Assign them to the Extractor Function.

You can assign customization function models to the following extractor function:

1. GTT relevance function of AOT for tracked process tracking
2. GTT relevance function of Event Type for event tracking
3. Planned Event Extractors
4. Control Parameter Extractors
5. Info Parameter Extractors(optional)
6. Tracking ID Extractors
7. Event Data Extractors
8. AOT ID Extractors

Please select one category above, create the extractor function and assign the corresponding modules.

For customization of GTT relevance and AOT ID, you need to enable *Determine by Function* option.

For customization of Tracking ID Type, you need to enable *Check Function(Function Module)* option.

Extractor	Description
510_WRF_CONTR_01	Control Parameters for Purchase Order (Seasonal Procurement)
CONTR_PARAM_DELIV	Selection of Control parameters for Deliveries in Shipment
OBP10_DELIV	Selection of CPs for Delivery - Outbound Delivery Visibility Process
OBP10_HU_IN_DLV	Selection of CPs for HUs in Delivery - Outbound Delivery Visibility Process
OCB10_CONTAINER	Selection of CPs for Containers in Ocean Carrier Booking Process
OCB10_ORDER	Selection of CPs for Booking Orders in Ocean Carrier Booking Process
ODT20_TOR	Selection of Control Parameters - Transportation Execution Visib. Proc.
ODT30_INS	Selection of Cntrl Parameters - Instruction Execution Visibility Procces
ODT40_TOR	Selection of Control Parameters - Transportation Execution Visib. Proc.
PCM10_ITEM	Selection of CPs for Purchase Order Item - Procurement Visibility Process
PMF10_NOTIF	Selection of CPs for Notification - Production Malfunction Visibility Process
PMF10_ORDER	Selection of CPs for Manuf. Order - Production Malfunction Visibility Process
RES30_CPARAM	Selection of Control Parameters - Resource Tracking Visibility Process
SNC10_MSGIN	Control Parameter Extractor for SNC Messages
SNC10_PURORD	Control Parameter Extractor for SNC Purchase Order
SNC10_RPLORD	Control Parameter Extractor for SNC Replenishment Order
TRA10_DELIV	Selection of CPs for Deliveries in Road Shipment - Transp. Visibility Process
TRA10_ROADSEA	Selection of CPs for Road/Sea Shipment - Transp. Visibility Process
ZGTT_OBP10_DELIV	Selection of CPs for Delivery - Outbound Delivery Visibility Process
ZGTT_OTE_DE_HDR	Control Parameter Extractor for Outbound Delivery Header
ZGTT_OTE_DE_ITEM	Control Parameter Extractor for Outbound Delivery Item
ZGTT_OTE_SHP_HDR	Control Parameter Extractor for Shipment Header
ZGTT_OTE_SO_HDR	Control Parameter Extractor for Sales Order Header
ZPOF_GTT_OTE_DL_HDR	Control Parameter Extractor for Inbound Delivery Header
ZPOF_GTT_OTE_DL_ITEM	Control Parameter Extractor for Inbound Delivery Item
ZPOF_GTT_OTE_PO_HDR	Control Parameter Extractor for Purchasing Order Header
ZPOF_GTT_OTE_PO_ITEM	Control Parameter Extractor for Purchasing Order Item
ZPOF_GTT_OTE_SH_HDR	Control Parameter Extractor for Shipment Header
ZSST_GTT_OTE_FO_HDR	Control Parameter Extractor for Freight Order

4: Sample Codes for Track Purchase Orders App

4-1 To support the Track Purchase Orders App, the sample codes cover the following cases by function group ZPOF_GTT:

Category	Business Process Type	Function Module Name	Description
Control Parameter Extractors	ESC_DELIV	ZPOF_GTT_OTE_DL_HDR	Control Parameter Extractor for Inbound Delivery Header
Control Parameter Extractors	ESC_DELIV	ZPOF_GTT_OTE_DL_ITEM	Control Parameter Extractor for Inbound Delivery Item
Control Parameter Extractors	ESC_PURORD	ZPOF_GTT_OTE_PO_HDR	Control Parameter Extractor for Purchasing Order Header
Control Parameter Extractors	ESC_PURORD	ZPOF_GTT_OTE_PO_ITEM	Control Parameter Extractor for Purchasing Order Item
Control Parameter Extractors	ESC_SHIPMT	ZPOF_GTT_OTE_SH_HDR	Control Parameter Extractor for Shipment Header
Event Data Extractors	ESC_MATDOC	ZPOF_GTT_EE_DL_ITEM_GR	Actual event PO Item Goods Receipt
Event Data Extractors	ESC_DELIV	ZPOF_GTT_EE_DL_ITEM_PA	Actual event PO Item Put Away
Event Data Extractors	ESC_DELIV	ZPOF_GTT_EE_DL_ITEM_PKNG	Actual event PO Item Packing
Event Data Extractors	ESC_PURORD	ZPOF_GTT_EE_PO_ITEM_CONF	Actual event PO Item Confirmation
Event Data Extractors	ESC_PURORD	ZPOF_GTT_EE_PO_ITEM_DEL	Actual event PO Item Deletion
Event Data Extractors	ESC_MATDOC	ZPOF_GTT_EE_PO_ITEM_GR	Actual event PO Item Goods Receipt
Event Data Extractors	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_ARR	Actual event Shipment Header Arrival
Event Data Extractors	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_CI	Actual event Shipment Header Check In
Event Data Extractors	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_DEP	Actual event Shipment Header Departure
Event Data Extractors	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_LE	Actual event Shipment Header Load End
Event Data Extractors	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_LS	Actual event Shipment Header Load Start
Planned Event Extractors	ESC_DELIV	ZPOF_GTT_EE_DL_ITEM	Selection of EEs for Inbound Delivery Item - Procurement Visibility Process
Planned Event Extractors	ESC_PURORD	ZPOF_GTT_EE_PO_HDR	Selection of EEs for Purchasing Order Header - Procurement Visibility Process
Planned Event Extractors	ESC_PURORD	ZPOF_GTT_EE_PO_ITEM	Selection of EEs for Purchasing Order Item - Procurement Visibility Process
Planned Event Extractors	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR	Selection of EEs for Shipment Header - Procurement Visibility Process
Tracking ID Extractors	ESC_DELIV	ZPOF_GTT_OTE_DL_ITEM_TID	Tracking ID Extractor for Inbound Delivery Item
Tracking ID Extractors	ESC_PURORD	ZPOF_GTT_OTE_PO_ITEM_TID	Tracking ID Extractor for Purchasing Order Item
Tracking ID Extractors	ESC_SHIPMT	ZPOF_GTT_OTE_SH_HDR_TID	Tracking ID Extractor for Shipment Header

4: Sample Codes for Track Purchase Orders App

4-2 To support the Track Purchase Orders App, the sample codes cover the following cases by function group ZPOF_GTT:

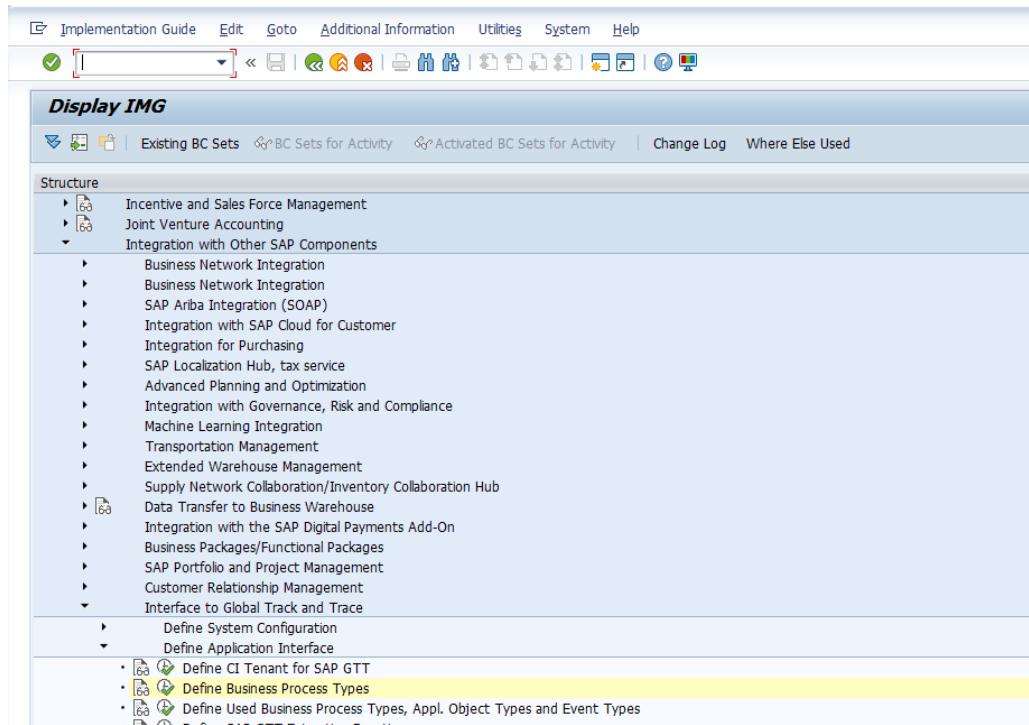
Category	Business Process Type	Function Module Name	Description
GTT relevance function of AOT	ESC_DELIV	ZPOF_GTT_OTE_DL_HDR_REL	Appl. Object Type Relevance for Inbound Delivery Header
GTT relevance function of AOT	ESC_DELIV	ZPOF_GTT_OTE_DL_ITEM_REL	Appl. Object Type Relevance for Inbound Delivery Item
GTT relevance function of AOT	ESC_PURORD	ZPOF_GTT_OTE_PO_HDR_REL	Appl. Object Type Relevance for Purchasing Order Header
GTT relevance function of AOT	ESC_PURORD	ZPOF_GTT_OTE_PO_ITEM_REL	Appl. Object Type Relevance for Purchasing Order Item
GTT relevance function of AOT	ESC_SHIPMT	ZPOF_GTT_OTE_SH_HDR_REL	Appl. Object Type Relevance for Shipment Header
GTT relevance function of Event Type	ESC_MATDOC	ZPOF_GTT_EE_DL_ITEM_GR_REL	Relevance function for Actual event PO Item Goods Receipt
GTT relevance function of Event Type	ESC_DELIV	ZPOF_GTT_EE_DL_ITEM_PA_REL	Relevance function for Actual event PO Item Put Away
GTT relevance function of Event Type	ESC_DELIV	ZPOF_GTT_EE_DL_ITEM_PKNG_REL	Relevance function for Actual event PO Item Packing
GTT relevance function of Event Type	ESC_PURORD	ZPOF_GTT_EE_PO_ITEM_CONF_REL	Relevance function for Actual event PO Item Confirmation
GTT relevance function of Event Type	ESC_PURORD	ZPOF_GTT_EE_PO_ITEM_DEL_REL	Relevance function for Actual event PO Item Deletion
GTT relevance function of Event Type	ESC_MATDOC	ZPOF_GTT_EE_PO_ITEM_GR_REL	Relevance function for Actual event PO Item Goods Receipt
GTT relevance function of Event Type	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_ARR_REL	Relevance function for Actual event Header Arrival
GTT relevance function of Event Type	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_CI_REL	Relevance function for Actual event Header Check In
GTT relevance function of Event Type	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_DEP_REL	Relevance function for Actual event Header Departure
GTT relevance function of Event Type	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_LE_REL	Relevance function for Actual event Header Load End
GTT relevance function of Event Type	ESC_SHIPMT	ZPOF_GTT_EE_SH_HDR_LS_REL	Relevance function for Actual event Header Load Start
Cross TP Update Function	ESC_PURORD	ZPOF_GTT_CTP_DL_TO_PO	Cross TP Update from Delivery to Purchase Order
Cross TP Update Function	ESC_DELIV	ZPOF_GTT_CTP_SH_TO_DL	Cross TP Update from Shipment to Delivery

5: Available Contexts for the Extractors' Modules

5-1: In Display IMG page, click
Integration with Other SAP Components -> Interface to Global Track and Trace -> Define Application Interface

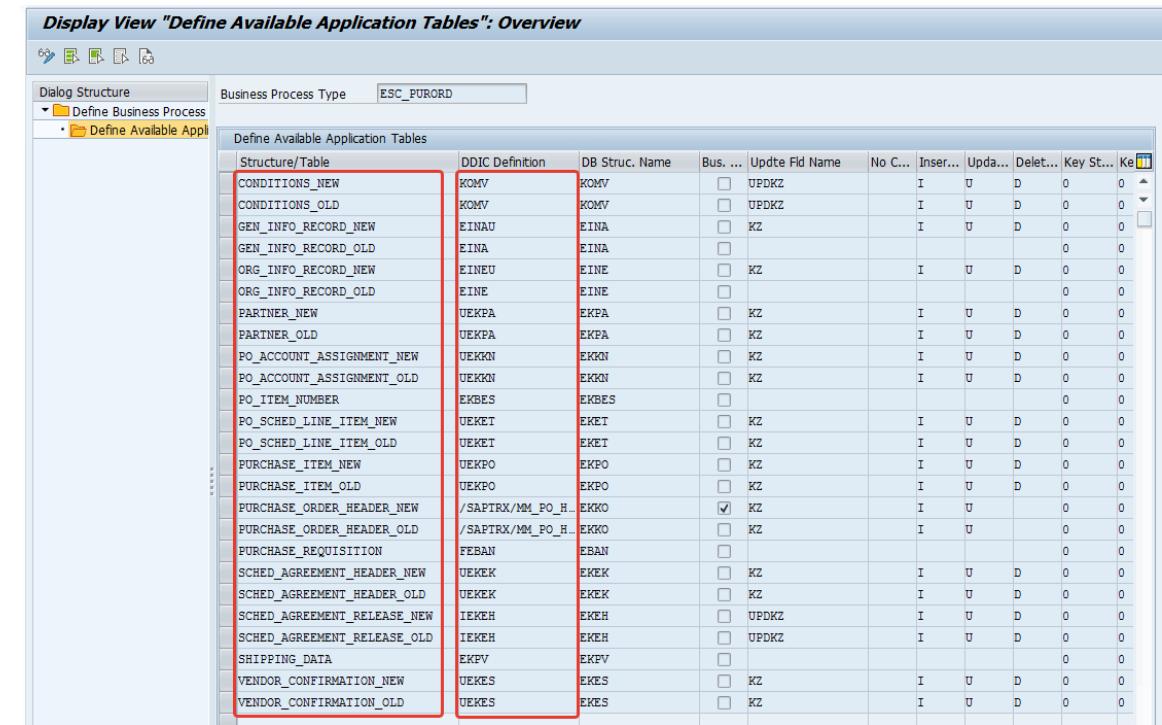
5-2: Choose activity **Define Business Process Types**

5-3: Please select the Business Process Types to find all the context tables and their structure info.



The screenshot shows the SAP Display IMG interface. The navigation bar at the top includes: Implementation Guide, Edit, Goto, Additional Information, Utilities, System, Help. Below the bar, there's a toolbar with various icons. The main area is titled "Display IMG" and shows a tree structure of SAP components. The "Structure" section is expanded, showing "Incentive and Sales Force Management", "Joint Venture Accounting", "Integration with Other SAP Components" (which is selected), and many other options like "Business Network Integration", "SAP Ariba Integration (SOAP)", etc. At the bottom of the tree, under "Define Application Interface", three items are listed: "Define CI Tenant for SAP GTT", "Define Business Process Types" (which is highlighted with a yellow background), and "Define Used Business Process Types, Appl. Object Types and Event Types".

Display View "Define Available Application Tables": Overview



This screenshot shows the SAP Display View "Define Available Application Tables" with the title "Overview". The view is filtered for the business process type "ESC_PURORD". The table lists various application tables along with their DDIC definitions, database structures, and update fields. The first 20 rows of the table are highlighted with a red border. The columns include:

Structure/Table	DDIC Definition	DB Struc. Name	Bus. ...	Updt Fld Name	No C...	Inser...	Upda...	Delet...	Key St...	Ke
CONDITIONS_NEW	KOMV		<input type="checkbox"/>	UPDKZ	I	U	D	0	0	
CONDITIONS_OLD	KOMV		<input type="checkbox"/>	UPDKZ	I	U	D	0	0	
GEN_INFO_RECORD_NEW	EINAU	EINA	<input type="checkbox"/>	KZ	I	U	D	0	0	
GEN_INFO_RECORD_OLD	EINA	EINA	<input type="checkbox"/>					0	0	
ORG_INFO_RECORD_NEW	EINEU	EINE	<input type="checkbox"/>	KZ	I	U	D	0	0	
ORG_INFO_RECORD_OLD	EINE	EINE	<input type="checkbox"/>					0	0	
PARTNER_NEW	UEKPA	EKPA	<input type="checkbox"/>	KZ	I	U	D	0	0	
PARTNER_OLD	UEKPA	EKPA	<input type="checkbox"/>	KZ	I	U	D	0	0	
PO_ACCOUNT_ASSIGNMENT_NEW	UEKKN	EKKN	<input type="checkbox"/>	KZ	I	U	D	0	0	
PO_ACCOUNT_ASSIGNMENT_OLD	UEKKN	EKKN	<input type="checkbox"/>	KZ	I	U	D	0	0	
PO_ITEM_NUMBER	EKBES	EKBES	<input type="checkbox"/>					0	0	
PO_SCHED_LINE_ITEM_NEW	UEKET	EKET	<input type="checkbox"/>	KZ	I	U	D	0	0	
PO_SCHED_LINE_ITEM_OLD	UEKET	EKET	<input type="checkbox"/>	KZ	I	U	D	0	0	
PURCHASE_ITEM_NEW	UEKPO	EKPO	<input type="checkbox"/>		I	U	D	0	0	
PURCHASE_ITEM_OLD	UEKPO	EKPO	<input type="checkbox"/>	KZ	I	U	D	0	0	
PURCHASE_ORDER_HEADER_NEW	/SAPTRX/MM_PO_H	EKKO	<input checked="" type="checkbox"/>	KZ	I	U	D	0	0	
PURCHASE_ORDER_HEADER_OLD	/SAPTRX/MM_PO_H	EKKO	<input type="checkbox"/>	KZ	I	U	D	0	0	
PURCHASE_REQUSITION	FEBAN	EBAN	<input type="checkbox"/>					0	0	
SCHED_AGREEMENT_HEADER_NEW	UEKEK	EKEK	<input type="checkbox"/>	KZ	I	U	D	0	0	
SCHED_AGREEMENT_HEADER_OLD	UEKEK	EKEK	<input type="checkbox"/>	KZ	I	U	D	0	0	
SCHED_AGREEMENT_RELEASE_NEW	IEKEH	EKEH	<input type="checkbox"/>	UPDKZ	I	U	D	0	0	
SCHED_AGREEMENT_RELEASE_OLD	IEKEH	EKEH	<input type="checkbox"/>	UPDKZ	I	U	D	0	0	
SHIPPING_DATA	EKPV	EKPV	<input type="checkbox"/>					0	0	
VENDOR_CONFIRMATION_NEW	UEKES	EKES	<input type="checkbox"/>	KZ	I	U	D	0	0	
VENDOR_CONFIRMATION_OLD	UEKES	EKES	<input type="checkbox"/>	KZ	I	U	D	0	0	

6: Coding Tips in the GTT Relevance Function Modules

To customize the GTT relevance function modules, key points are as below:

1. Make sure that the Main / Master tables are following the configuration of corresponding AOT or Event Type.
2. Add customization logics to determine the output parameters *E_RESULT*.

See sample code of function: *ZPOF_GTT_OTE_PO_ITEM_REL*

The image shows two SAP ABAP development environments side-by-side. On the left is the 'Function Builder: Display ZPOF_GTT_OTE_PO_ITEM_REL' window, which displays the source code for the function module. On the right is the 'ABAP Editor: Display Include LZPOF_GTTD20' window, which displays the source code for the include module.

Function Builder: Display ZPOF_GTT_OTE_PO_ITEM_REL

```
DATA: lt_app_objects TYPE txas_appobj_ctabs,
      lo_udm_message TYPE REF TO cx_udm_message,
      ls_bapiret    TYPE bapiret2.

      lt_app_objects = VALUE #( ( i_app_object ) ).

TRY.
  e_result = lcl_ef_performer->check_relevance(
    is_definition = VALUE #(
      maintab = lif_pof_constants->cs_tabledef-po_item_new
      mastertab = lif_pof_constants->cs_tabledef-po_header_new
      NEW lcl_factory_po_item( )
    )
    iv_bo_factory = i_apps
    i_apps
    is_app_obj_types = i_app_obj_types
    it_all_appl_tables = i_all_appl_tables
    it_app_objects = lt_app_objects
  ).

CATCH cx_udm_message INTO lo_udm_message.
  lcl_tools->get_errors_log(
    EXPORTING
      io_udm_message = lo_udm_message
      iv_appsys = i_appsys
    IMPORTING
      es_bapiret = ls_bapiret
  ).

  " add error message
  APPEND ls_bapiret TO c_logtable.

  " throw corresponding exception
CASE lo_udm_message->textid.
  WHEN lif_ef_constants->cs_errors-stop_processing.
    RAISE stop_processing.
  WHEN lif_ef_constants->cs_errors-table_determination.
    RAISE parameter_error.
ENDCASE.

ENDTRY.
```

ABAP Editor: Display Include LZPOF_GTTD20

```
METHOD lif_bo_reader-check_relevance.
  " 1. Basic check of main table which shall be following
  " the AOT configuration
  " 2. Check that only 1 PO type is relevance for GTT,
  " which could be the standard PO type: NB
  " 3. If it's CREATING PO, always flag TRUE
  " 4. If it's UPDATING PO, check whether there is any
  " change for all the above fields or not, comparing
  " their NEW / OLD value pairs
  " 5. Don't need to consider DELETING PO, which will be
  " considered by standard logic of EM framework and
  " extractors cannot impact this case

  rv_result = lif_pof_constants->cs_condition-false.

  " is_app_object-maintabdef = lif_pof_constants->cs_tabledef-po_item_new AND
  IF lcl_po_tools->is_appropriate_po_type( ir_ekko = is_app_object-mastertabref ) = abap_true AND
  lcl_po_tools->is_appropriate_po_item( ir_ekpo = is_app_object-maintabref ) = abap_true AND
  is_object_changed( is_app_object = is_app_object ) = abap_true.

  CASE is_app_object-update_indicator.
    WHEN lif_ef_constants=>cs_change_mode-insert.
      rv_result = lif_ef_constants=>cs_condition-true.
    WHEN lif_ef_constants=>cs_change_mode-update OR
        lif_ef_constants=>cs_change_mode-undefined.
      lif_ef_constants=>cs_change_mode-undefined.
      rv_result = lcl_tools->are_structures_different(
        ir_data1 = lif_bo_reader-get_data(
          is_app_object = is_app_object
        )
        ir_data2 = lif_bo_reader-get_data_old(
          is_app_object = is_app_object
        )
      ).
    ENDCASE.
  ENDIF.

ENDMETHOD.

METHOD lif_bo_reader-get_data.
  FIELD-SYMBOLS: <ls_item> TYPE ts_po_item.
  rr_data = NEW ts_po_item( ).

  ASSIGN rr_data->* TO <ls_item>.
  IS CONNECT CHANGED.
```

7: Coding Tips in the Tracking ID Function Modules

To customize the Tracking ID function modules, key points are as below:

1. Make sure that the Main / Master tables are following the configuration of corresponding AOT.
2. Add customization logics to fill the output table *E_TRACKIDDATA*.
3. The Tracking ID Type need to be the same as the definition in the process type of model in Manage Models app.
4. GTT v2 accepts delta transport for tracking IDs, which means that only the newly-created / changed / deleted tracking IDs shall be filled, while the ones without change need to be ignored in the logic.
5. The tracking ID for its own process type needs to be filled for each process update.
6. In case of tracking ID deletion, the field *ACTION* shall be filled with 'D'.

See sample code of function: *ZPOF_GTT_OTE_PO_ITEM_TID*

Function Builder: Display ZPOF_GTT_OTE_PO_ITEM_TID

```

Function Group: ZPOF_GTT
Function: ZPOF_GTT_OTE_PO_ITEM_TID
Attributes: Import, Export, Changing, Tables, Exceptions, Source Code

19 DATA: lo_udm_message TYPE REF TO cx_udm_message,
20      ls_bapiret TYPE bapiret2.
21
22 TRY.
23   lcl_ef_performer->get_track_id_data(
24     EXPORTING
25       is_definition = VALUE #((
26         maintab = lif_pof_constants->cs_tabledef-po_item_new
27         mastertab = lif_pof_constants->cs_tabledef-po_header_new
28         io_bo_factory = NEW lcl_factory_po_item( )
29         iv_appsyst = i_appsyst
30         is_app_obj_types = i_app_obj_types
31         it_all_appl_tables = i_all_appl_tables
32         it_app_type_ctrl_tabs = i_app_type_ctrl_tabs
33         it_app_objects = i_app_objects
34       IMPORTING
35         et_track_id_data = e_trackiddata[])
36   .
37
38   CATCH cx_udm_message INTO lo_udm_message.
39   lcl_tools->get_errors_log(
40     EXPORTING
41       io_udm_message = lo_udm_message
42       iv_appsyst = i_appsyst
43     IMPORTING
44       es_bapiret = ls_bapiret .
45
46     " add error message
47     APPEND ls_bapiret TO e_logtable.
48
49     " throw corresponding exception
50     CASE lo_udm_message->textid.
51       WHEN lif_ef_constants->cs_errors_stop_processing.
52         RAISE stop_processing.
53       WHEN lif_ef_constants->cs_errors_table_determination.
54         RAISE table_determination_error.
55     ENDCASE.
56   ENDTRY.

```

ABAP Editor: Display Include LZPOF_GTTD20

```

Include: LZPOF_GTTD20 Active
METHOD lif_bo_reader-get_track_id_data.
  FIELD-SYMBOLS: <ls_ekpo> TYPE ukpo,
                  <lt_ekes> TYPE lif_pof_types->tt_uekes.
  DATA(lv_tzone) = lcl_tools->get_system_time_zone( ).

  DATA(lr_ekes) = mo_ef_parameters->get_appl_table(
    iv_tabledef = lif_pof_constants->cs_tabledef-po_vend_conf_new).

  ASSIGN is_app_object-maintabref-> TO <ls_ekpo>.

  CLEAR: et_track_id_data[].

  IF <ls_ekpo> IS ASSIGNED.
    et_track_id_data = VALUE #( ((
      appsys = mo_ef_parameters->get_appsyst()
      appobjtype = is_app_object-appobjtype
      appobjid = is_app_object-appobjid
      trxcod = lif_pof_constants->cs_trxcod-po_position
      trxid = #( <ls_ekpo>-ebeln )`<ls_ekpo>-ebelp ) )
    start_date = lcl_tools->get_system_date_time()
    end_date = lif_ef_constants->cv_max_end_date
    timzon = lv_tzone
    msrid = space
    ) ).

  IF <ls_ekpo>-kz = lif_ef_constants->cs_change_mode-insert.
    et_track_id_data = VALUE #( BASE et_track_id_data (
      appsys = mo_ef_parameters->get_appsyst()
      appobjtype = is_app_object-appobjtype
      appobjid = is_app_object-appobjid
      trxcod = lif_pof_constants->cs_trxcod-po_number
      trxid = #( <ls_ekpo>-ebeln )
      start_date = lcl_tools->get_system_date_time()
      end_date = lif_ef_constants->cv_max_end_date
      timzon = lv_tzone
      msrid = space
    ) ).

  ENDFIF.

```

8: Coding Tips in the Control Parameter Function Modules

To customize the Control Parameter function modules, key points are as below:

1. Make sure that the Main / Master tables are following the configuration of corresponding AOT.
2. Add customization logics to fill the output table *E_CONTROL_DATA*.
3. GTT v2 asks for full transport for all the control parameters, which means that all the fields needs to be extracted in all cases, no matter whether their values have been changed.
4. To fill up the composition (table) fields defined in Manage Models app, use the parameter field *PARAMINDEX* to specify the line number. If the field is empty, GTT regards it as a simple flat field.
5. To clear a composition, fill the key field using invalid values, for which key attribute has been checked in Manage Model app. It's not recommended to fill a code list type field to clear a composition even if it's a key field.
6. The field with fixed name 'ACTUAL_BUSINESS_DATETIME' and 'ACTUAL_BUSINESS_TIMEZONE' are mandatory fields to be transported for event handling sequencing in GTT V2.
7. In Manage Models app, click tab *IDOC Integration* to map the parameter names and model field names.
8. For DATE or DATETIME fields, when the source value is initial like '00000000' '0000000000000000', then please ensure to only enable *PARAMNAME* and *PARAMINDEX* in the extractor code, *not enable VALUE* for IDOC sending.
9. For Amount field which has reference currency, please ensure to call BAPI 'BAPI_CURRENCY_CONV_TO_EXTERNAL' using the reference currency to make the amount tracked correctly by GTT v2. The BAPI will output the conversion result in 4 decimals as fixed, which needs additional rounding in the extractor if the corresponding field defined in the tracking model is of less than 4 decimals.

See sample code of function: *ZPOF_GTT_OTE_PO_ITEM*

8: Coding Tips in the Control Parameter Function Modules

Fields mapping is set up in Manage Models app in IDOC Integration section:

pof Active

Purchase Order Fulfillment

Namespace: com.lbngttsamples.gtt.app.pof Correlation Level: 4

Tracked Process Field Type Pool Event Type Pool Code List **IDOC Integration** Visibility Provider Integration Planned Event Extension Event to Action

Tracked Process: PurchaseOrder ▼ Integration Switch: ON

Tracked Process Mapping

ERP Object Type: Others Application Object Type: ZPOF_GTT_AC_HD

Tracked Process / Events (1)

Name	IDOC	Event Code
Tracked Process		
PurchaseOrderEvent	E1EHPAO	

Fields

Field	IDOC Segment	IDOC Field
purchaseOrderNo	E1EHPCP	YN_PO_NUMBER
supplierId	E1EHPCP	YN_PO_SUPPLIER_ID
plannedDeliveryDate	E1EHPCP	YN_PO_DELIVERY_DATE
netValue	E1EHPCP	YN_PO_NET_VALUE
currency	E1EHPCP	YN_PO_CURRENCY
incotermsVersion	E1EHPCP	YN_PO_INCOTERMS_VERSION
incoterms	E1EHPCP	YN_PO_INCOTERMS
incotermsLocation	E1EHPCP	YN_PO_INCOTERMS_LOCATION

8: Coding Tips in the Control Parameter Function Modules

Main logic of Purchase Order Item is implemented in class LCL_BO_READER_PO_ITEM

Function Module ZPOF_GTT_OTE_PO_ITEM active

Attributes Import Export Changing Tables Exceptions Source Code

```
19 DATA: lo_udm_message      TYPE REF TO cx_udm_message,
20      ls_bapiret        TYPE bapiret2.
21
22 TRY.
23   lcl_ef_performer->get_control_data(
24     EXPORTING
25       is_definition      = VALUE #(
26         maintab           = lif_pof_constants->cs_tabledef-po_item_new
27         mastertab          = lif_pof_constants->cs_tabledef-po_header_new )
28   io_bo_factory      = NEW lcl_factory_po_item( )
29   iv_appsps          = i_appsps
30   is_app_obj_types   = i_app_obj_types
31   it_all_appl_tables = i_all_appl_tables
32   it_app_type_cntl_tabs = i_app_type_cntl_tabs
33   it_app_objects     = i_app_objects
34
35 CHANGING
36   ct_control_data    = e_control_data[] ).
37
38 CATCH cx_udm_message INTO lo_udm_message.
39   lcl_tools->get_errors_log(
40     EXPORTING
41       io_udm_message = lo_udm_message
42       iv_appsps     = i_appsps
43     IMPORTING
44       es_bapiret    = ls_bapiret ).
45
46 " add error message
47 APPEND ls_bapiret TO e_logtable.
48
49 " throw corresponding exception
50 CASE lo_udm_message->textid.
51   WHEN lif_ef_constants->cs_errors-stop_processing.
52     RAISE stop_processing.
53   WHEN lif_ef_constants->cs_errors-table_determination.
54     RAISE table_determination_error.
55 ENDCASE.
56
57 ENTRY.
58 ENDFUNCTION.
```

ABAP Editor: Display Include LZPOF_GTTD20

Repository Browser

Include LZPOF_GTTD20 Active

```
19841 METHOD lif_bo_reader~get_data.
19842   FIELD-SYMBOLS: <ls_item>      TYPE ts_po_item.
19843
19844   rr_data  = NEW ts_po_item( ).
19845
19846   ASSIGN rr_data->* TO <ls_item>.
19847
19848   fill_item_from_ekko_struct(
19849     EXPORTING
19850       ir_ekko      = is_app_object-mastertabref
19851     CHANGING
19852       cs_po_item  = <ls_item> .
19853
19854   fill_item_from_ekpo_struct(
19855     EXPORTING
19856       ir_ekpo      = is_app_object-mastertabref
19857     CHANGING
19858       cs_po_item  = <ls_item> .
19859
19860   fill_item_from_eket_table(
19861     EXPORTING
19862       ir_ekpo      = is_app_object-mastertabref
19863       ir_eket      = mo_ef_parameters->get_appl_table(
19864         iv_tabledef = lif_pof_constants->cs_tabledef-po_sched_new )
19865     CHANGING
19866       cs_po_item  = <ls_item> .
19867
19868   fill_item_from_ekes_table(
19869     EXPORTING
19870       ir_ekpo      = is_app_object-mastertabref
19871       ir_ekes      = mo_ef_parameters->get_appl_table(
19872         iv_tabledef = lif_pof_constants->cs_tabledef-po_vend_conf_new )
19873     CHANGING
19874       cs_po_item  = <ls_item> .
19875
19876   fill_item_location_types(
19877     CHANGING
19878       cs_po_item  = <ls_item> .
19879
19880 ENDMETHOD.
```

9: Coding Tips in the Planned Event Function Modules

To customize the Planned Event function modules, key points are as below:

1. Make sure that the Main / Master tables are following the configuration of corresponding AOT.
2. Add customization logics to fill the output table *E_EXPEVENTDATA*.
3. GTT v2 asks for full transport for all the planned events, which means that all the events needs to be extracted in all cases, no matter whether their values have been changed.
4. The field *MILESTONE* is mandatory to be transported.
5. The field *EVT_EXP_DATETIME* is optional, but need to be filled with relevant time zone *EVT_EXP_TZONE* together if it needs to be transported.
6. The field *LOC_ID1* is optional, but need to be filled with relevant location type *LOCTYPE* together if it needs to be transported. The values for field *LOCTYPE* are limited by *Manage Locations* app in GTT V2.
7. The field *LOCID2* is mandatory to specify the stop ID (match key) in case of shipment tracking.

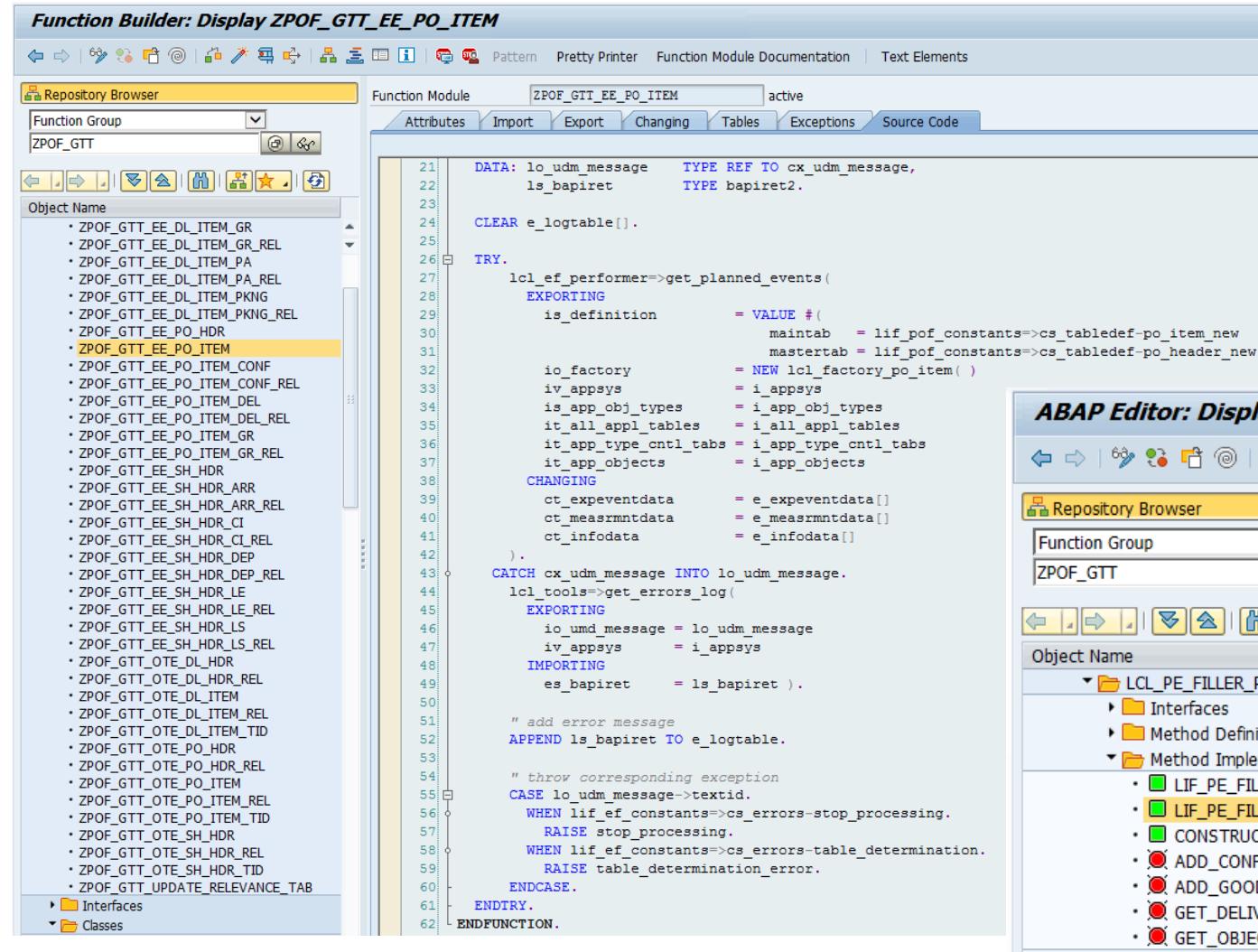
See sample code of function: *ZPOF_GTT_EE_PO_ITEM*

The screenshot shows a SAP Fiori application interface for managing Planned Event Function Modules. At the top, the module name is "pof" with a status of "Active". Below that, the description is "Purchase Order Fulfillment" and the namespace is "com.lbngttsamples.gtt.app.pof". The correlation level is set to 4. A red box highlights the "IDOC Integration" tab in the top navigation bar. The main content area displays a table titled "Tracked Process Mapping" with one row: "Tracked Process: PurchaseOrderItem" and "ERP Object Type: Others". Below this is a section titled "Tracked Process / Events (5)" with a table showing five rows of tracked processes and their corresponding IDOC codes. A red box highlights the entire table. The last section, also highlighted with a red box, is titled "Event Types" and lists four event types with their respective IDOC codes and descriptions.

Name	IDOC	Event Code
Tracked Process		
PurchaseOrderItemEvent	E1EHPAO	
Event Types		
ConfirmationEvent	E1EVMHDR02	CONFIRMATION
GoodsReceipt	E1EVMHDR02	GOODS_RECEIPT
DeletionEvent	E1EVMHDR02	DELETION
UndeletionEvent	E1EVMHDR02	UNDELETION

9: Coding Tips in the Planned Event Function Modules

Main logic of Purchase Order Item Planned Events is implemented in class LCL_PE_FILLER_PO_ITEM

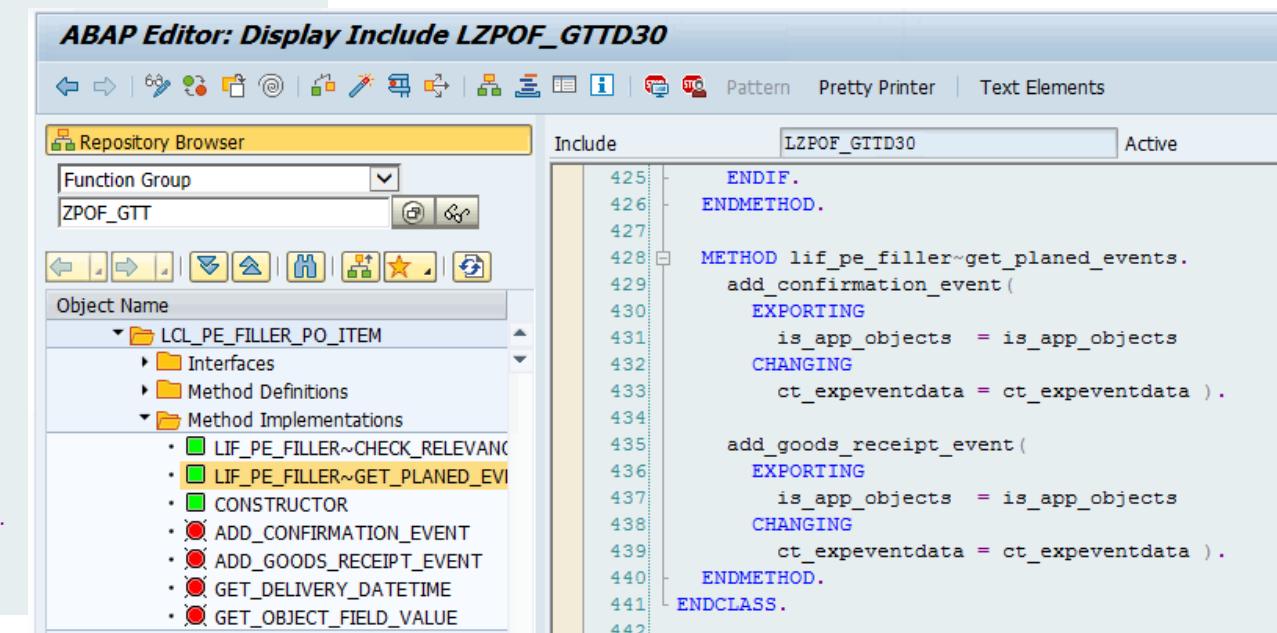


```
DATA: lo_udm_message    TYPE REF TO cx_udm_message,
      ls_bapiret     TYPE bapiret2.

CLEAR e_logtable[].

TRY.
  lcl_ef_performer->get_planned_events(
    EXPORTING
      is_definition      = VALUE #(
        maintab   = lif_pof_constants->cs_tabledef-po_item_new
        mastertab = lif_pof_constants->cs_tabledef-po_header_new )
    IMPORTING
      io_factory       = NEW lcl_factory_po_item( )
      iv_appsyst      = i_appsyst
      is_app_obj_types = i_app_obj_types
      it_all_appl_tables = i_all_appl_tables
      it_app_type_cntl_tabs = i_app_type_cntl_tabs
      it_app_objects   = i_app_objects
    CHANGING
      ct_expeventdata  = e_expeventdata[]
      ct_measrmntdata = e_measrmntdata[]
      ct_infodata      = e_infodata[])
  .
  CATCH cx_udm_message INTO lo_udm_message.
    lcl_tools->get_errors_log(
      EXPORTING
        io_udm_message = lo_udm_message
        iv_appsyst     = i_appsyst
      IMPORTING
        es_bapiret     = ls_bapiret ).
  "
    add error message
  APPEND ls_bapiret TO e_logtable.

  "
    throw corresponding exception
  CASE lo_udm_message->txid.
    WHEN lif_ef_constants=>cs_errors-stop_processing.
      RAISE stop_processing.
    WHEN lif_ef_constants=>cs_errors-table_determination.
      RAISE table_determination_error.
  ENDCASE.
ENDTRY.
ENDFUNCTION.
```



```
ENDIF.
ENDMETHOD.

METHOD lif_pe_filler~get_planed_events.
  add_confirmation_event(
    EXPORTING
      is_app_objects = is_app_objects
    CHANGING
      ct_expeventdata = ct_expeventdata ).

  add_goods_receipt_event(
    EXPORTING
      is_app_objects = is_app_objects
    CHANGING
      ct_expeventdata = ct_expeventdata ).

ENDMETHOD.
ENDCLASS.
```

10: Coding Tips in the Event Data Function Modules

To customize the Event Data function modules, key points are as below:

1. Make sure that the Main / Master tables are following the configuration of corresponding Event Type.
2. Add customization logic to fill the output table *CT_TRACKINGHEADER*, *CT_TRACKLOCATION*, *C_EVENTID_MAP*.
3. If the event has user-defined fields in Manage Models app, fill the table *CT_TRACKPARAMETERS*.
4. If the event has reference table information, fill the table *CT_TRACKREFERENCES*.
5. The field *CT_TRACKINGHEADER-SRCCOD*, *SRCID*, *SRCTX* is used for event reason transport.
6. In Manage Models app, click tab IDOC Integration to map the user-defined parameter names and model field names.

See sample code of function: *ZPOF_GTT_EE_PO_ITEM_CONF*

10: Coding Tips in the Event Data Function Modules

To set up mapping of event type user-defined parameters, go to the IDOC Integration section of Manage Models app, select corresponding event type and set values of IDOC Field:

pof Active

Purchase Order Fulfillment

Namespace: com.lbngttsamples.gtt.app.pof Correlation Level: 4

Tracked Process Field Type Pool Event Type Pool Code List **IDOC Integration** Visibility Provider Integration Planned Event Extension Event to Action

Tracked Process: PurchaseOrderItem ▼ Integration Switch: ON

Tracked Process Mapping

ERP Object Type: Others Application Object Type: ZPOF_GTT_AC_ITEM

Tracked Process / Events (5)

Name	IDOC	Event Code
Tracked Process		
PurchaseOrderItemEvent	E1EHPAO	
Event Types		
ConfirmationEvent	E1EVMPAR02	CONFIRMATION
GoodsReceipt	E1EVMPAR02	GOODS_RECEIPT
DeletionEvent	E1EVMPAR02	DELETION
UndeletionEvent	E1EVMPAR02	UNDELETION

Fields

Field	IDOC Segment	IDOC Field
quantity	E1EVMPAR	QUANTITY
confirmType	E1EVMPAR	CONFIRM_TYPE

10: Coding Tips in the Event Data Function Modules

Main logic of Purchase Order Item Confirmation event is implemented in class LCL_AE_FILLER_PO_ITEM_CONF

Function Module ZPOF_GTT_EE_PO_ITEM_CONF active

Attributes Import Export Changing Tables Exceptions Source Code

```
59: DATA: lo_udm_message      TYPE REF TO cx_udm_message,
60:         ls_bapiret        TYPE bapiret2.
61:
62: TRY.
63:   lcl_ae_performer->get_event_data(
64:     EXPORTING
65:       is_definition      = VALUE #((
66:         maintab          = lif_pof_constants->cs_tabledef-po_item_new
67:         masterstab        = lif_pof_constants->cs_tabledef-po_header_new )
68:       io_ae_factory     = NEW lcl_ae_factory_po_item_conf( )
69:       iv_appsyst        = i_appsyst
70:       is_event_type     = i_event_type
71:       it_all_appl_tables = i_all_appl_tables
72:       it_event_type_cntl_tabs = i_event_type_cntl_tabs
73:       it_events         = i_events
74:     CHANGING
75:       ct_eventid_map    = c_eventid_map[]
76:       ct_trackingheader = ct_trackingheader[]
77:       ct_tracklocation  = ct_tracklocation[]
78:       ct_trackreferences = ct_trackreferences[]
79:       ct_trackparameters = ct_trackparameters[]
80:   .
81: CATCH cx_udm_message INTO lo_udm_message.
82:   lcl_tools->get_errors_log(
83:     EXPORTING
84:       io_udm_message = lo_udm_message
85:       iv_appsyst     = i_appsyst
86:     IMPORTING
87:       es_bapiret     = ls_bapiret .
88:
89:   " add error message
90:   APPEND ls_bapiret TO ct_logtable.
91:
92:   " throw corresponding exception
93:   CASE lo_udm_message->textid.
94:     WHEN lif_ef_constants=>cs_errors-stop_processing.
95:       RAISE stop_processing.
96:     WHEN lif_ef_constants=>cs_errors-table_determination.
97:       RAISE event_data_error.
98:   ENDCASE.
99: ENDTRY.
100: ENDFUNCTION.
```

ABAP Editor: Display Include LZPOF_GTTD40

Repository Browser

Object Name

- ZPOF_GTT
- Interfaces
- Classes
 - LCL_AE_FACTORY
 - LCL_AE_FACTORY_DL_ITEM_GR
 - LCL_AE_FACTORY_DL_ITEM_PA
 - LCL_AE_FACTORY_DL_ITEM_PKNG
 - LCL_AE_FACTORY_PO_ITEM_CONF
 - LCL_AE_FACTORY_PO_ITEM_DEL
 - LCL_AE_FACTORY_PO_ITEM_GR
 - LCL_AE_FACTORY_SH_HEADER_ARR
 - LCL_AE_FACTORY_SH_HEADER_CI
 - LCL_AE_FACTORY_SH_HEADER_DEP
 - LCL_AE_FACTORY_SH_HEADER_LE
 - LCL_AE_FACTORY_SH_HEADER_LS
 - LCL_AE_FILLER_DL_ITEM_GR
 - LCL_AE_FILLER_DL_ITEM_PA
 - LCL_AE_FILLER_DL_ITEM_PKNG
 - LCL_AE_FILLER_PO_ITEM_CONF
- Method Definitions
- Method Implementations
 - LIF_AE_FILLER~CHECK_RELEVANCY
 - LIF_AE_FILLER~GET_EVENT_DATA
 - CONSTRUCTOR
 - GET_CONFIRMATION_QUANTITY
 - GET_CONFIRMATION_QUANTITY_
 - HAS_CHANGES
 - IS_APPROPRIATE_CONF CONTRC
 - IS_APPROPRIATE_CONF_TYPE
- Attributes
- LCL_AE_FILLER_PO_ITEM_DEL
- LCL_AE_FILLER_PO_ITEM_GR
- LCL_AE_FILLER_SH_HEADER_ARR
- LCL_AE_FILLER_SH_HEADER_BH

Include LZPOF_GTTD40 Active

```
98: METHOD lif_ae_filler~get_event_data.
99:   DATA (lv_difference) = get_confirmation_quantity_diff(
100:     is_events = is_events ).
101:
102:   ct_trackingheader = VALUE #( BASE ct_trackingheader (
103:     language      = sy-langu
104:     trxid        = lcl_po_tools->get_tracking_id_po_item(
105:       ir_ekpo = is_events-maintabref )
106:     trxcod      = lif_pof_constants->cs_trxcod-po_position
107:     evtcnt      = is_events-eventid
108:     evtid        = lif_pof_constants->cs_milestone-po_confirmation
109:     evtdat      = sy-datum
110:     evttim      = sy-uzzeit
111:     evtzon      = lcl_tools->get_system_time_zone( )
112:   )).
113:
114:   ct_eventid_map = VALUE #( BASE ct_eventid_map (
115:     eventid     = is_events-eventid
116:     evtcnt      = is_events-eventid
117:   )).
118:
119:   ct_tracklocation = VALUE #( BASE ct_tracklocation (
120:     evtcnt      = is_events-eventid
121:     loccod      = lif_ef_constants->cs_loc_types-plant
122:     locidl      = lcl_tools->get_field_of_structure(
123:       ir_struct_data = is_events-maintabref
124:       iv_field_name = 'WERKS' )
125:   )).
126:
127:
128:   ct_trackparameters = VALUE #( BASE ct_trackparameters (
129:     evtcnt      = is_events-eventid
130:     param_name  = lif_pof_constants->cs_event_param-quantity
131:     param_value = lcl_tools->get_pretty_value( iv_value = lv_difference )
132:   )).
133:
134:
135:   ct_trackparameters = VALUE #( BASE ct_trackparameters (
136:     evtcnt      = is_events-eventid
137:     param_name  = lif_pof_constants->cs_event_param-confirm_type
138:     param_value = lif_pof_constants->cs_relevance-ebtyp
139:   )).
140: ENDMETHOD.
```

11: Enhancement Codes for Cross-process Tracking

The Track Purchase Orders app asks for cross-processes tracking, which is used in below cases:

1. When the inbound delivery item process is updated and transported to GTT, the preceding purchasing order item process needs to be updated and transported to GTT.
2. When the shipment process is updated and transported to GTT, the preceding inbound delivery and item process, and their planned events needs to be updated and transported to GTT.

IMPORTANT: To enable cross-process tracking, please update the below sample codes after downloading:

1. Update Purchase Order Item AOT type Mask in Method GET_AOTYPE_RESTRICTIONS of LCL_CTP_SENDER_DL_TO_PO_ITEM
2. Update Inbound Delivery Header and Item AOT type Mask in Method GET_AOTYPE_RESTRICTIONS of LCL_CTP_SENDER_SH_TO_DL_HEAD and LCL_CTP_SENDER_SH_TO_DL_ITEM

ABAP Editor: Display Include LZPOF_GTTD80

```
338 CLASS lcl_ctp_sender_dl_to_po_item IMPLEMENTATION.
339   METHOD get_aotype_restrictions.
340     et_aotype = VALUE #((
341       low   = 'ZPOF_GTT_*_ITEM'
342       option = 'CP'
343       sign   = 'I'
344     )).
345   ENDMETHOD.
346
347   METHOD get_instance.
348     DATA(lt_trk_obj_type) = VALUE tt_trk_obj_type(
349       ( lif_ef_constants=>cs_trk_obj_type-esc_deliv )
350       ( lif_ef_constants=>cs_trk_obj_type-esc_purord )
351     ).
352 
```

ABAP Editor: Display Include LZPOF_GTTD80

```
1551 CLASS lcl_ctp_sender_sh_to_dl_head IMPLEMENTATION.
1552   METHOD get_aotype_restrictions.
1553     et_aotype = VALUE #((
1554       low   = 'ZPOF_GTT_*_DL HD'
1555       option = 'CP'
1556       sign   = 'I'
1557     )).
1558   ENDMETHOD.
1559
1560   METHOD get_instance.
1561     DATA(lt_trk_obj_type) = VALUE tt_trk_obj_type(
1562       ( lif_ef_constants=>cs_trk_obj_type-esc_shipmt )
1563       ( lif_ef_constants=>cs_trk_obj_type-esc_deliv )
1564     ).
```

11: Enhancement Codes for Cross-process Tracking

The cross-process tracking scenarios cover the following:

Delivery Item -> Purchase Order Item:

- 1\ Inbound Delivery Item Composition (Full Transport)
 - Case: Inbound Delivery Item Create / Delete
 - Case: Inbound Delivery Create / Delete

Shipment -> Inbound Delivery and Inbound Delivery Item:

- 1\ Tracking ID (Delta Transport)
 - Case: Shipment Create / Delete with Delivery
 - Case: Shipment Assign / Unassign Delivery
- 2\ Shipment Composition (Full Transport)
 - Case: Shipment Create / Delete with Delivery
 - Case: Shipment Assign / Unassign Delivery
- 3\ Planned Event in Delivery (Full Transport)
 - Case: Shipment Create / Delete with Delivery / with stage
 - Case: Shipment Assign / Unassign Delivery / with stage
 - Case: Stage Assign / Unassign Delivery
 - Case: Stage Insert / Delete
 - Case: Stage Location Update
 - Case: Stage Planned Datetime Update
- 4\ Planned Event in Delivery Item (Full Transport)
 - Case: Shipment Create / Delete with Delivery / with stage
 - Case: Shipment Assign / Unassign Delivery / with stage
 - Case: Stage Assign / Unassign Delivery
 - Case: Stage Insert / Delete
 - Case: Stage Location Update
 - Case: Stage Planned Datetime Update

12: Known Issues

1. Planned Event Extension not enabled

By now, on ERP side, the EXTENSION segment of process IDOC is not enabled for the planned event part, which means that user cannot make the user-defined fields based on the planned event level in Manage Models.

The workaround is to take use of Control Parameter's segment in IDOC and make the field mapping on process level in Manage Models.

2. IDOC sequencing issue

By now, on ERP side, when the user is reporting actual events while creating the process, the IDOCs will be sent out of sequence. For example, entering a PICK quantity and saving the new delivery in ERP will generate a PICK event IDOC and a delivery order IDOC. If the event IDOC approaches GTT prior to the order IDOC, which will lead into processing failure.

This issue will be covered in short future.

Follow us



www.sap.com/contactsap

© 2020 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

See www.sap.com/copyright for additional trademark information and notices.

THE BEST RUN  SAP[®]