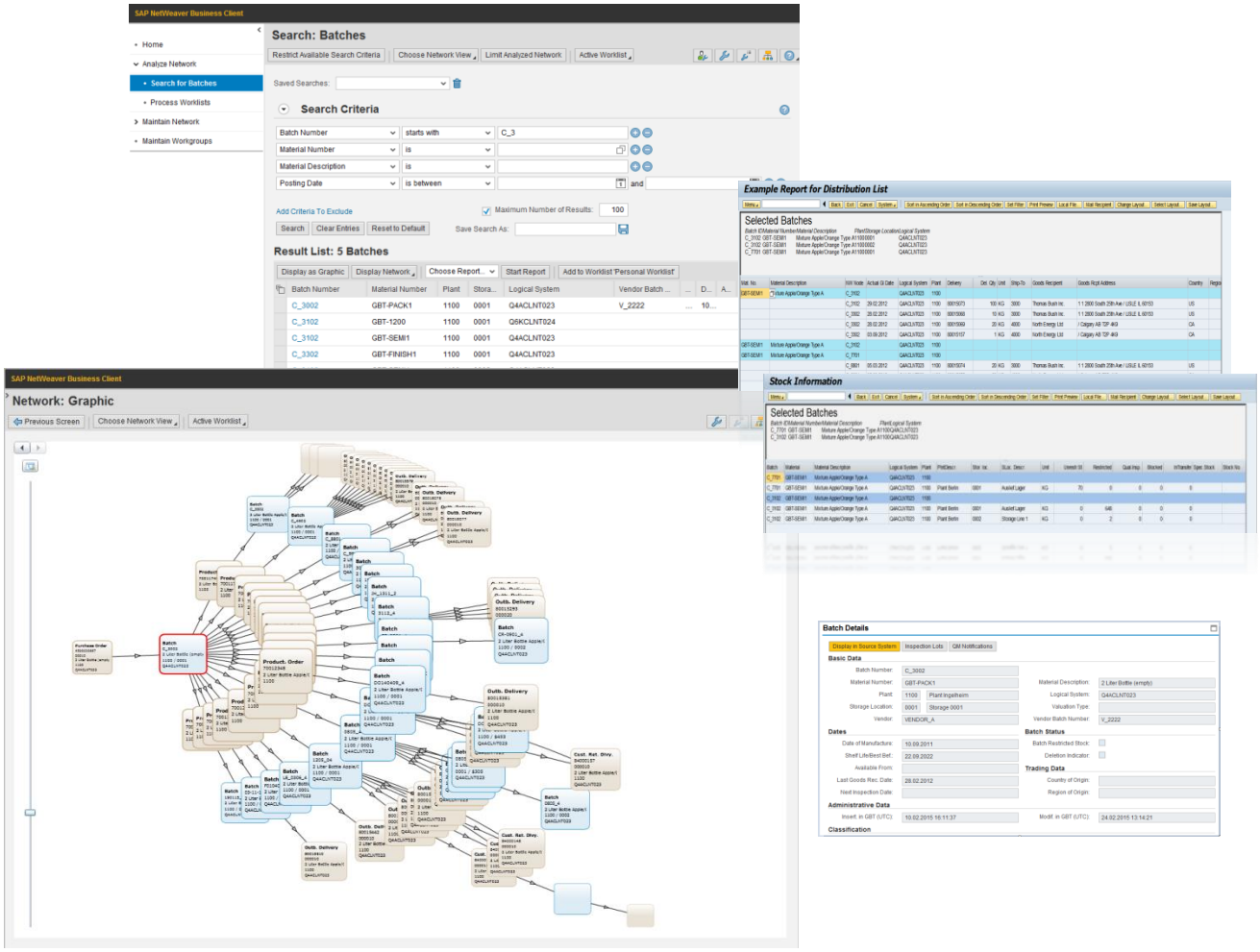
Extensibility of SAP Global Batch Traceability 2.0 (SAP GBT)



Document version: 2.0 (SP07) V12 – 2016-19-09
PUBLIC

Disclaimer

In the following, you find SAP-owned information which is not part of the SAP product documentation. By using this documentation, YOU AGREE that unless expressly stated otherwise in your agreements with SAP, the content of this document is not part of SAP product documentation and that you may not infer any product documentation claims against SAP based on this information.

Contents

1 Introduction

SAP Global Batch Traceability offers a repository for the genealogy of batches, a partial quantity for a material respectively product.

The genealogy has the relation (aka GBT-Events) between different batches and / or business documents which occurs in the supply chain.

The focus of this document is to describe how the GBT genealogy and the business objects shown in the genealogy graph can be extended.

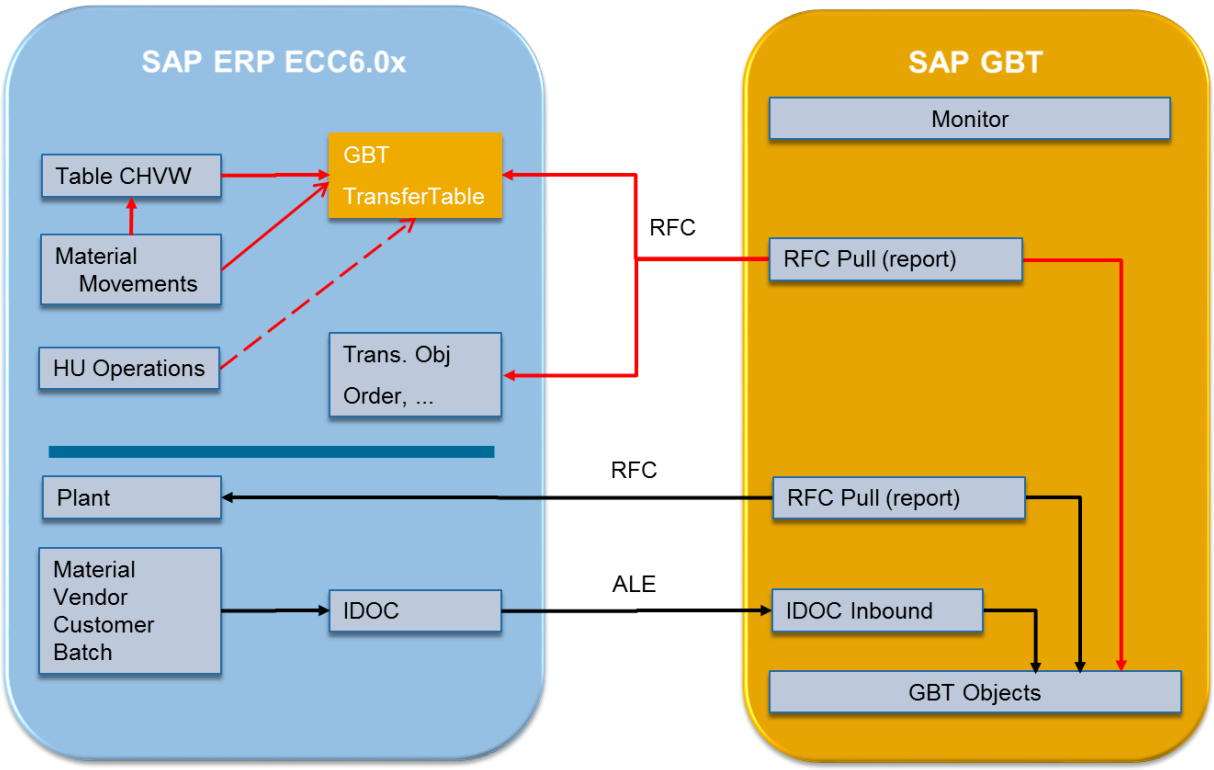
Tip: See also Note 1257033 - Cookbook: Modification/enhancement for standard SAP system.

Please also fell free to check GBT on SCN.

1.1 Architecture Overview of GBT

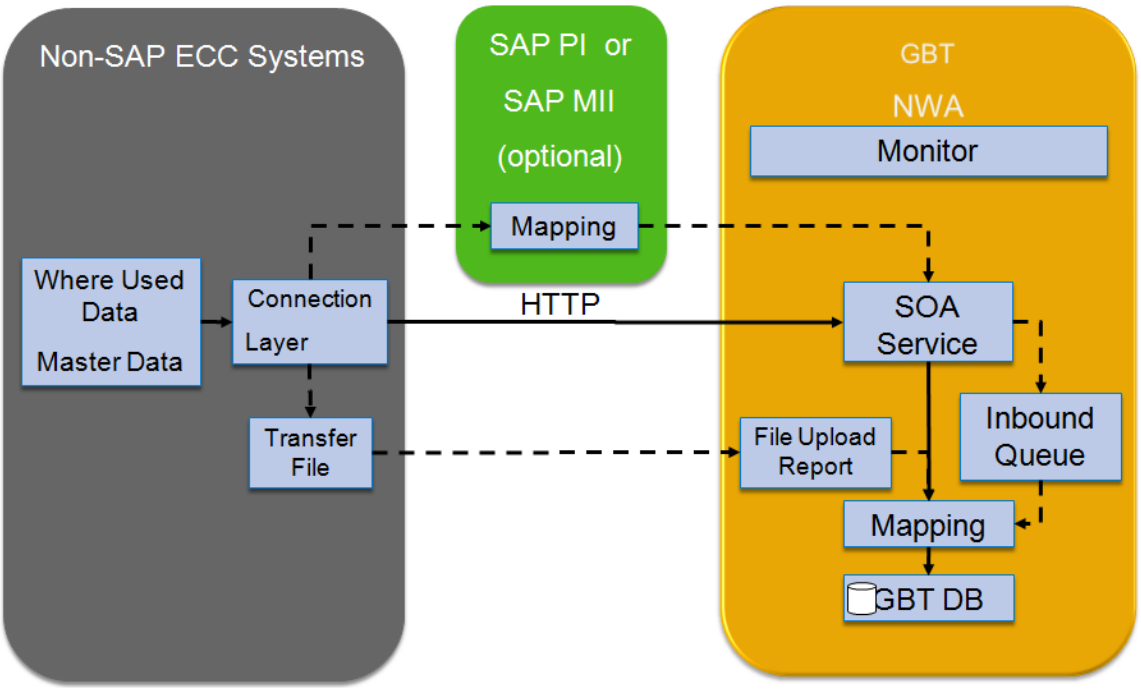
GBT supports integration to ERP600 out of the box. The GBT business documents are tailored to store the GBT relevant attributes of an ERP Business document. Specific attributes which are part of a customer SAP-ERP implementation naturally not supported out of the box and may be needed to be added to GBT.

The following diagram shows the principals integration with SAP ERP:



- Note: Events only triggered if the Plant is registered to send GBT-Events. To generate GBT-Events for HUs an additionally registration is needed. The HU-Events are separated and independent from Material Movements generated within BAdi BADI_HU_SAVE.

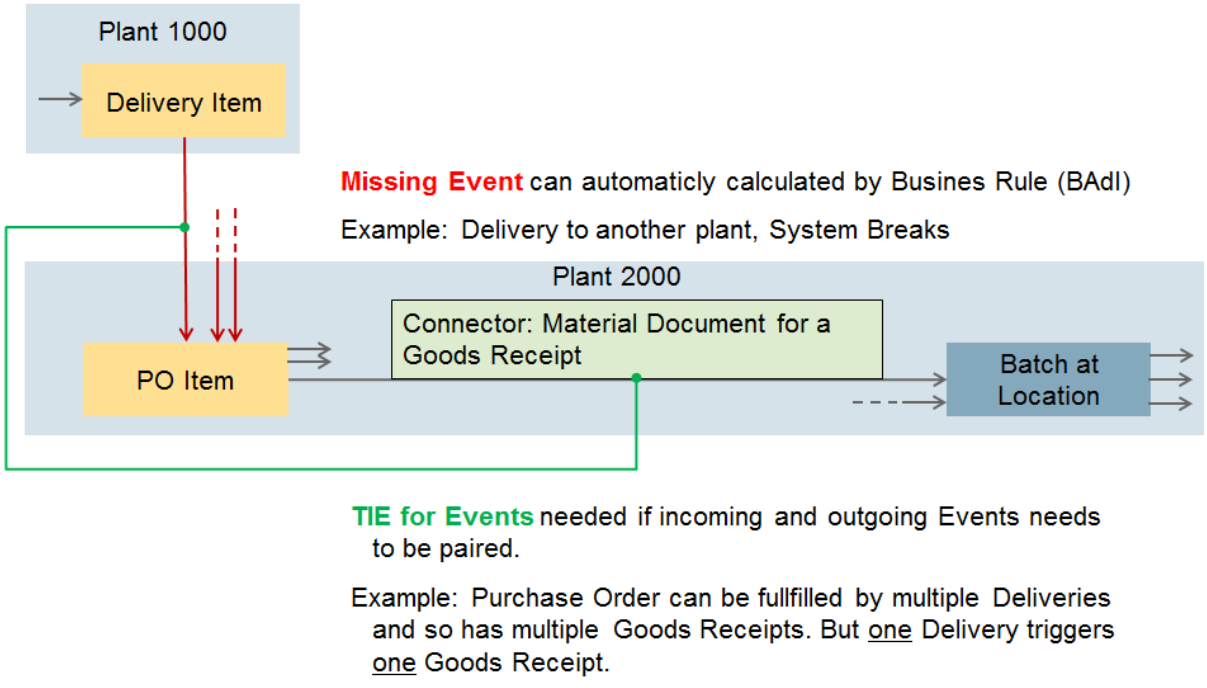
If a non-SAP-ERP system should be connected to GBT it is quite likely that the ERP business documents not fit. Therefore it is also possible to extent the object model in GBT.



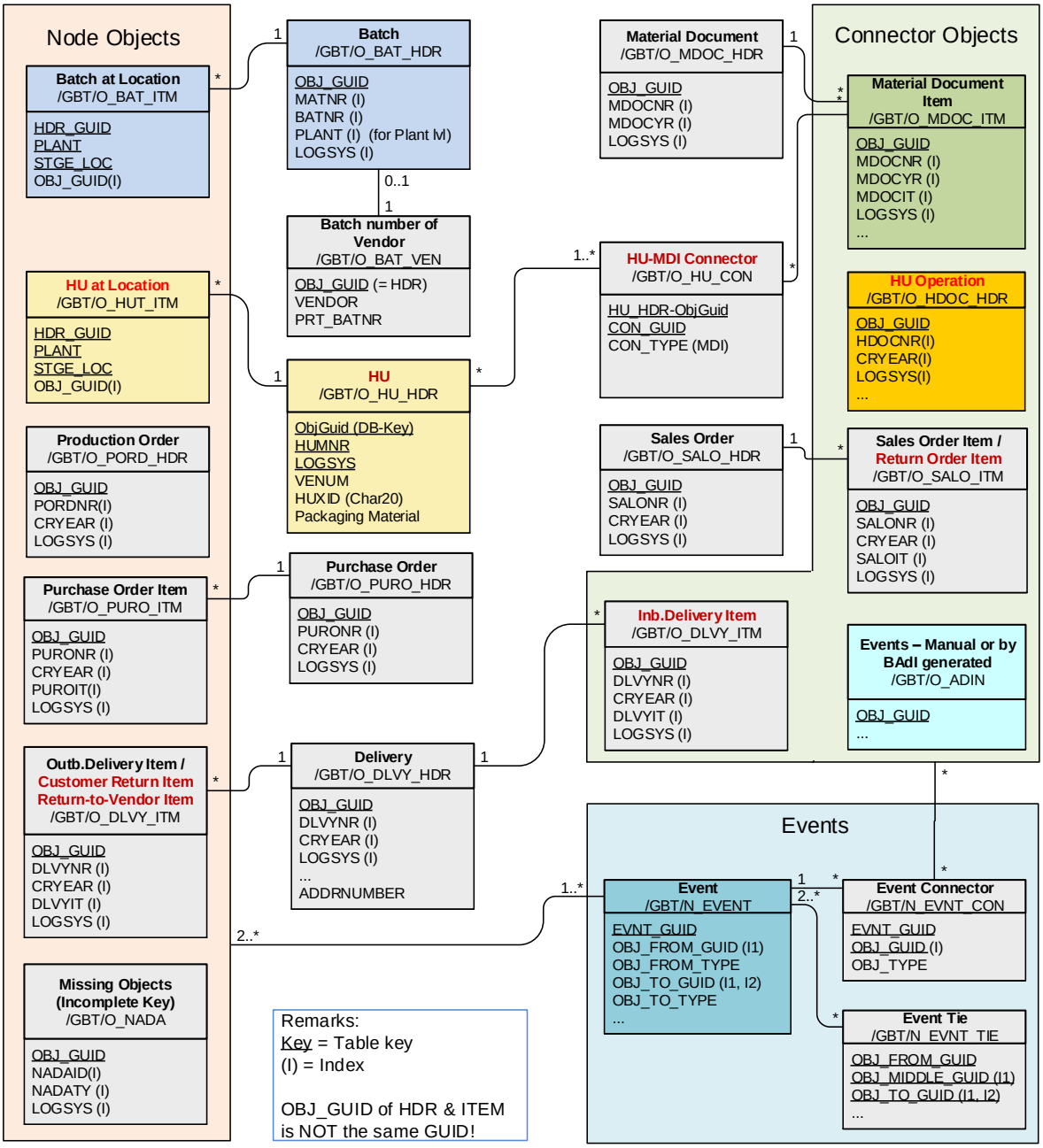
1.2 Data model of GBT

The following picture shows the structure of GBT Events. A GBT Event is a directed relation between 2 objects. The trigger for the Event is a connector object which links the 2 objects together.

Remark: In some cases, especially for object Production Order or Purchase Orders Items, several incoming and outgoing events are possible. There can be a relation between one incoming event to one outgoing event and not to others. To model this, the events can be tied together.



Database tables of GBT



Master Data (all Reuse)

ABA Product

Business Partner

Classification for Batches

Customizing (without Reuse Tables)

Objects for Network
/GBT/C_TRC_OBJ

Logsys Connected to GBT
/GBT/C_SYS_DEST

/GBT/C_...

NEW Object Types GBT2.0 (FP2) (different Obj.Types can share same Table (like modeled in ERP))

Only in case the Business Function for HUs is switched on the tables /GBT/HU_CON and /GBT/HU_HDR are used.

1.3 Overview of Extensibility in GBT

1.3.1 Object Extensibility

Batch master extensibility is provided by using classification.

Master Data Implementation in GBT reuses the standard implementation of the ABA-layer (nothing additional for GBT): Product, Business partner

If more data from IDOC are needed to store in GBT you can

- for Product (MATMAS, /GBT/MATMAS) implement the BAdI /GBT/EX_IL_MATMAS where the Enhancements can be done.
- In any case it is also possible to create a new IDOC Inbound function module (existing GBT FM can be copied and enhanced) and exchange the GBT-IDOC-FM by the new IDOC-FM.

For an extension of ALE / IDOC for inbound processing in GBT see here.

Extensibility of other Business documents (used in the GBT Network):

Field extensibility of existing business documents supported by customer includes.

New business documents needs to be registered in GBT with their access class.

Integration from ERP needs to be extended by implementing a BAdI in GBT and offer an Access Class in ERP and enhance data of GBT-Events in ERP or also trigger new GBT-Events.

The integration to a non SAP ECC System can be done via SOA Services as well as File import. (For SOA Service which can create events between any objects used in the GBT Network).

Out-of-the box this is possible for the GBT-Events which build up the genealogy network.

To get details of the Business documents from a non-SAP ERP system is also task of a GBT implementation project.

The UI is based on Floor Plan Mangager (FPM) for WebDynpro ABAP and Microsoft Silverlight for the Graphic. For changes of the WebDynpro UI mainly the FPM configuration needs to be adapted. The SPI-Framework is used to feed the FPM UI (Generic UI Building Block (GUIBB)). The graphic is embedded into WebDynpro (WebDynpro Island for Silverlight) and is able to displays all objects ut-of-the box if they are integrated in the Network.

1.3.2 Process Extensibility

Add new Reports, which can be called from Batch Search and Worklist via FPM Launchpad.

Add Actions to Worklist which can be triggered from UI.

Navigate to other UIs / ERP Transactions via FPM Launchpad

2 Batch Master

The GBT implementation of the Batch Master uses the classification (CA-CL) as well as the ERP Batch implementation. The classification is the main enhancement possibility for additional Batch attributes. This includes a fully integration into the GBT Bach search functionality as well as visibility in the Batch Details UI.

The classification master data (classes and characteristics), the classification of batches and the batches itself are integrated from ERP with ALE / IDOC.

2.1 Mapping of Class and Characteristic for Classification

If multiple systems provide their characteristics and classes it may be needed to distinct same names which are defined in different Systems differently. To do so the GBT offers the BAdI /GBT/EX_CLF_MAPPING

(Customizing Path: Cross-Application Components -> SAP Global Batch Traceability -> Business Add-Ins (BAdIs) for SAP GBT -> Integration -> BAdI: Mapping of Class and Characteristic for Classification)F

The 'LOBM'-Characteristics are delivered as 'SAP_GBT': To get them use report /GBT/R_COPY_STANDARD_CHARC to copy them into the current client.

2.2 Extend the Batch Master in GBT

To see the additional fields in the Table view or Batch Detail screen a new configuration of the ABAP-WebDynpro and Floor Plan Manager (FPM) configuration is needed. For details see here.

To add fields to the Batch master it is recommended to use classification.

2.2.1 Extend by Classification

Classification is the standard way to enhance a Batch with additional attributes (aka characteristics).

The Batch Characteristics are only displayed in the Batch Detail screen. As a precondition to display them also in Graphic and/or Table-View the needed characteristics must be added to the Include CI_BAT_CHR.

For example: Assume the internal value for a characteristic, let say COLOUR, is '3' and the textual value is 'purple'. The field COLOUR (same name as the characteristic) needs to be added to CI_BAT_CHR and the data element should have the size to contain the textual value.

Instead of adding a new field to CI_BAT_CHR also a structure can be included which contains ONE characteristic. The GROUP name of this structure must be equal to the characteristic name. This approach is needed in two cases:

1. If the Characteristic name is not allowed as fieldname (e.g. the hyphen '-' is not allowed) a structure must be used. The structure contains only one field, with a freely defined name to get the textual value.

2. If beside the textual value also the internal value is needed. The 1st field of the structure has to be the text field and the 2nd field has to be the value field. The names of the structure and the fields in this structure can be freely defined e.g. structure ZMY_COLOUR with fields COLOUR_TXT and COLOUR_VAL. If the value has a Unit (or Currency) this must be the 3rd field.

2.2.2 Extension of Batch Table

2.2.2.1 Extension of Batch Table /GBT/O_BAT_HDR

There are also in some rare cases it may be needed to extend the Batch Master table in GBT. Therefore the database table /GBT/O_BAT_HDR has an include structure (customer Include):

CI_BAT_ATT. This structure has to be created with the missing attributes. To fill these attributes in ALE Integration the segment E1BPBNCOMZ in ERP (and the corresponding segment /GBT/E1BPBNCOMZ in GBT) can be extended by using WE31. If the attributes of the segment /GBT/E1BPBNCOMZ exists with the same name in CI_BAT_ATT they will be copied by function module /GBT/IL_IDOC_IN_BATMAS.

For an extension of ALE / IDOC for inbound processing in GBT see here.

2.2.2.2 Extension of Batch Table /GBT/O_BAT_ITM

The database table /GBT/O_BAT_ITM contains the Plant /Storage location specific entries. GBT has by standard no attributes on this level. But there is a 'customer include': CI_BAT_ITM_ATT. To fill this from ERP the BAdI /GBT/EX_IL_LOAD_DETAIL can be implemented for object type BAT.

2.3 Search for Batches

Batches at a location can be search with quick search by batch number or with the standard 'Search for Batches' which allows a combination of several attributes. If an attribute is not offered as search criteria to 'Search for Batches' this can be added.

The Batch Search Screen, Global Batch Traceability 2.0 provides the following search criteria:

- *Batch*
  - Batch Number (BATNR)
  - Material Number (MATNR)
  - Plant (PLANT)
  - Logical System (LOGSYS)
  - Vendor Batch Number (PRT_BATNR)
- *Material*
  - Material Description (MATERIAL_DESCR)
- *Product Group*

  Any Product Group defined is added dynamically

- *Handling Unit (Only offered f Business function /GBT/HU_SFWS is switched on)*
  - Handling Unit (HUMNR)
  - Packaging Material (PACK_MAT)
  - Warehouse (WHSENUMBER)
  - WM Transfer Order (WMTONR)
- *Delivery*
  - Ship-To Party (SHIP_TO)
  - Sold-To Party (SOLD_TO)
  - Actual Goods Issue Date (GI_DATE)
- *Material Document*
  - Posting Date (PSTNG_DATE)
  - Movement Type (MOVE_TYPE)
- *Characteristics*

  Added dynamically (can be defined by creating batch class GBT_SEL_CRITERIA as described in SAP note 1870447)

2.3.1 Add new Search Criteria to 'Search for Batches'

Creating an implicit enhancement at the end of method /GBT/CL_NW_MP->ADD_SEL_CRIT_COMPLEX_SEARCH and add some coding to define the additional search criteria.

2.3.1.1 Prefix for Search Criteria

To enable the query to identify the object to which the field belongs to, a prefix is used in the technical name of the new search criterion. The prefix is used to define the object and therefore the query path for specific criteria.

The following prefixes (plus separator '%') have to be used for the different object types in the batch search:

- *Batch search attributes: SAP%*

Attributes of network objects used in search:

- *Handling Unit: HUX%*
- *Delivery: DLV%*
- *Material Document: MDI%*

… and other existing Object types (plus separator %)

Additional prefix which can be used for enhancements which needs customer specific selections

- *Material: MAT% (see method /GBT/CL_MAT_BO_UTIL-> ENHANCE_SEL_PARAM)*

Attention: It may be that some combinations of search criteria are not possible and the error message "`Selection path is too complex`" occurs.

See the example where the prefix 'SAP', separated by '%' from the fieldname of the criteria internal fieldname, is used for the batch field PROD_DATE in variable *ls_component-name*.

2.3.1.2 Sorting of Search Criteria

The sequence of the search criteria is defined first by the number in field *ls_component_detail-description_s* and then by the text in field *ls_component_detail-description_l*.

Each object type like batch or delivery has a number range for the first sort field *ls_component_detail-description_s* to group the search criteria:

- *Batch: 001 - 009*
- *Product Group: 010*
- *Handling Unit: 021 - 030*
- *Deliveries: 031 - 040*
- *Material Documents: 041 - 050*
- *Characteristics: 999*

Define the sequence of the search criteria either by assigning specific numbers, or by using the same number to sort alphabetically by field *ls_component_detail-description_l.*

2.3.1.3 Example how to enhance Batch Search Criteria

The following example describes how to add the field PROD_DATE (Date of Manufacture) to the batch search:

Enter the following coding in the enhancement before the statement ENDMETHOD:

```
* Batch: 'Date of Manufacture'
  CLEAR ls_component.
  CLEAR ls_component_detail.
```

```
  ls_component-name = 'SAP%PROD_DATE'.           "<Prefix>%<Field Name>
  ls_component-type = '/GBT/BAT_PROD_DATE'.      "Data element
  APPEND ls_component TO cs_criteria_def-components.

  ls_component_detail-name          = ls_component-name.
  ls_component_detail-description_s = '007'.     "1. Sort field -> Group
  ls_component_detail-
description_l = /gbt/cl_spi_util=>get_data_element_description( ls_compon
ent-type ).
  ls_component_detail-
options = /gbt/cl_spi_util=>fill_char_select_options( ).
  INSERT ls_component_detail INTO TABLE cs_criteria_def-
component_details.
```

2.3.2 Adapt batch specific search criteria

To adapt 'SAP%'-criteria just before used for batch search they can be checked and modified in Method /GBT/CL_BAT_BO->ENHANCE_SEL_PARAM. In case the criteria 'material number' needs to be adapted check also if the Method /GBT/CL_MAT_BO_UTIL-> ENHANCE_SEL_PARAM needs to be adapted.

2.3.3 Enhance Search result with additional Attributes

In case that additional attributes, which have no further relevance in other UIs, should be shown in the search result list, an additional enhancement option is possible.

The attributes needs directly added to the Node-Structure using an append structure (see here).

To set the attribute values the BAdI-Interface /PLMB/IF_EX_SPI_APPL_ACCESS in the Method AFTER_QUERY

2.3.4 Predefined Search Patterns

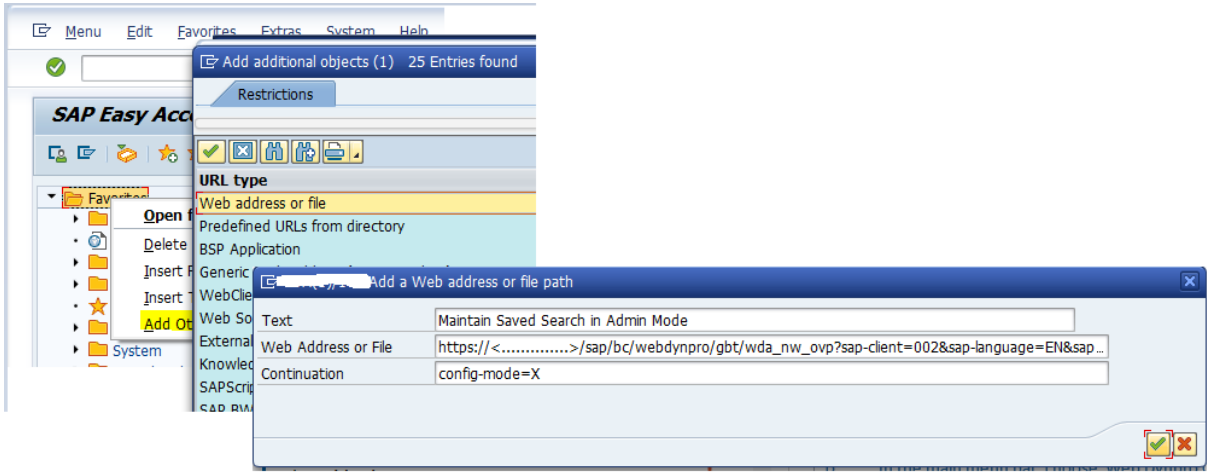
With NW7.40 and Support Level >= SP3 the FPM offers the possibility to store predefined Searches.

You need to run a WD-application explicitly in administrative mode

To do this, proceed as follows:

- Open transaction SE80.
- In the Object Navigator navigate to the relevant Applic. Configurationsfolder and double-click an application configuration inside it.
- In the main menu bar, choose Web Dynpro Configuration Test Execute in Administration Mode.
- The application is opened in administrator mode; check if the Customizing Mode banner is displayed above the title bar.

Tip: The URL used with Web Dynpro can be added as short cut to the Favorite of the Administrators SAP-GUI-Menu



How to add an URL as Favorite to the SAP-GUI-Menu

Remark: The URL is expected to be longer than one line => fill line “Web address or file” up to the end AND insert the trailing missing part into the line “Continuation”.

3 Network Objects of Batch Genealogy Network

The GBT Batch genealogy consists of Batches and business document such as Purchase Order, Production or Process Order or Outbound-Delivery. The business document items and Batches at a specific Plant/ Storage Location are the nodes in the network. The relations between these nodes based on GBT-Events which derived mainly from Material Documents Items as primary connector. Also supplemental connectors like Sales Order can be assigned to an Event to hold further information in GBT.

Remark: all the Objects in the Batch Genealogy Network are defined by an Object type and a GUID.

3.1 Extension of received GBT-Events (BAdI)

It may be needed to provide new Events in GBT which are not coming from an external system but which can be provided by GBT. An example is Cross System Scenario where a single Systems only provides its objects but he linkage (GBT-Event) can be calculated in GBT.

You can use example BAdI implementation /GBT/EX_IL_BATCH_TIE_X_STO to create additional event links across stock transfer order items to link received batch and issued batch by batch and material number. Furthermore, you can use example BAdI implementation /GBT/EX_IL_EVENT_ENHANCE to create additional events to link delivery item and purchase order item imported from different systems.

Terminology explanation:

Event:	Represents a goods movement booking step in the connected system. It connects two objects that are relevant for the network with a booking step. Example: You have a batch 4711 of product STEEL_123 for a delivery item 10 of a delivery 800001. You book a goods issue event at time 1 about 10 pieces and a goods issue event at time 2 about 50 pieces. As a result two events are created.
Event link:	A link between 2 relationships and connects 3 network objects

For details about the BAdI, check the documentation and example implementation of the BAdI /GBT/EX_IL_EVENT_ENHANCE (for an example of an implementation see section Important GBT Classes).

3.2 Extension of business documents

3.2.1 Field extension

Two types of field extensions are needed. Fields for object attributes which needs to be stored additional in the DB and fields for linked attributes which are already stored in GBT like e.g. a description for a field value.

3.2.1.1 Field extension with DB table enhancement

All Business documents can be extended by customer includes on header database table as well as on the item database table. To see the additional fields in the detailed screen a new WebDynpro configuration of the specific detailed screen needs to be created.

Database tables and the related customer include:

Business Documents	**Header**		**Item**	
	DB Table	Customer Include	DB table	Customer Include
Material Document	/GBT/O_MDOC_HDR	CI_MDOC_ATT	/GBT/O_MDOC_ITM	CI_MDOC_ITM_ATT
Purchase Order	/GBT/O_PURO_HDR	CI_PURO_ ATT	/GBT/O_PURO_ITM	CI_PURO_ITM_ATT
Production or Process Order	/GBT/O_PORD_HDR	CI_PORD_ATT	-	-
Delivery	/GBT/O_DLVY_HDR	CI_DLVY_ATT	/GBT/O_DLVY_ITM	CI_DLVY_ITM_ATT
Sales Order	/GBT/O_SALO_HDR	CI_SALO_ATT	/GBT/O_SALO_ITM	CI_SALO_ITM_ATT
Batch at a Location	-	-	/GBT/O_BAT_ITM	CI_BAT_ITM_ATT

Attention: Same named attributes in Customer Include for header table and item table are not allowed!

3.2.1.2 UI specific field extension without DB table enhancement

All Business documents can be extended without DB extension by an append-structure to the related 'Node'-structure used for UI. To see the additional fields in the detailed screen a new WebDynpro configuration of the specific detailed screen needs to be created.

To fill these new append fields additional logic can be placed in the read method /GBT/IF_TRC_OBJ~GET_UI_DATA_BY_GUID of the class /GBT/CL_TRC_BO_<...> as Implicit Enhancements.

Business Document	**DDIC Structure for UI**	**Internal used read method /GBT/IF_TRC_OBJ~GET_UI_DATA_BY_GUID of Class**
Batch	/GBT/S_BAT_NODE_STRUCT	/GBT/CL_TRC_BO_BAT
Batch Characteristics	CI_BAT_CHR	(see Remark to Batch Characteristic below)
Material Document Item	/GBT/S_MDIT_NODE_STRUCT	/GBT/CL_TRC_BO_MDIT
Purchase Order Item	/GBT/S_PURI_NODE_STRUCT	/GBT/CL_TRC_BO_PURI
Production or Process Order	/GBT/S_PORD_NODE_STRUCT	/GBT/CL_TRC_BO_PORD
Outb. Delivery Item	/GBT/S_DLIO_NODE_STRUCT	/GBT/CL_TRC_BO_DLIO
Sales Order Item	/GBT/S_SALI_NODE_STRUCT	/GBT/CL_TRC_BO_SALI

An **example for setting appended fields**.

Remark:

To extend UI specific structures and to fill them only for the WebUI purpose the BAdIs of Enhancement Spot /PLMB/ES_SPI , especial /PLMB/EX_SPI_METADATA and /PLMB/EX_SPI_APPL_ACCESS, can be used (example use case).

3.2.2 New business documents

All business documents shown in the Genealogy network needs an Entry in the Customizing.

(Customizing Path: Cross-Application Components -> SAP Global Batch Traceability -> Business Add-Ins (BAdIs) for SAP GBT -> Definition of Object Types -> Define Objects for Global Batch Traceability Network)

This customizing needs a class which provides the access to the business document persistency.

The following steps are needed:

1. Create the Database table(s) for the new business document

   The Purchase Order is an example in GBT with header and item table. For the Production Order only the header is relevant.

   Add the field OBJ_GUID with type /GBT/OBJ_GUID to each table.

   Remark: Business document often consist of a header and items. If only the header data should be used in the genealogy the item table is not needed (as for ERP production Order). An item-entry and its corresponding header-entry have different values for OBJ_GUID.

DB Table name	Purpose
ZGBT_O_<…>_HDR	Business Document Header table with field OBJ_GUID
ZGBT_O_<…>_ITM	Business Document Item table with field OBJ_GUID

   As Example see DB table /GBT/O_PURO_* .
   Remark: 'ZGBT' represents a non-SAP namespace and '<…>' is an abbreviation for the business document.

2. Create several structures and table types

   E.g. typically the following are needed

Structure name	Purpose
ZGBT_S_<…> _KEY	Key for Business Document inclusive field LOGSYS (if needed also year of creation)
ZGBT_S_<…>_NODE_STRUCT	Business Document Item with all Item and also header attributes .
ZGBT_S_<…>_UI_LABEL	ID for UI (Graphic) of the Object
ZGBT_S_<…>_UI_OVERVIEW	ID and main attributes for UI (Graphic)
ZGBT_S_<…>_MAINT	Business Document Maintenance
ZGBT_S_<…>_MAINT_ITM	Business Document Maintenance for Item

   E.g. see structures /GBT/S_PUR*

Table type name	Purpose
ZGBT_T_<…>_MAINT	Business Document Maintenance
ZGBT_T_<…>_MAINT_ITM	Business Document Maintenance for Item

   Tip: E.g. see table types /GBT/T_PUR*

3. Create a class (SE24) for the new business document which is used in the genealogy.
   This class needs to implements the interface /GBT/IF_TRC_OBJ (E.g. adopt a copy of an existing class like /GBT/CL_TRC_BO_PURI).
   /GBT/IF_TRC_OBJ defines methods which has untyped attributes for the business document import and export structures and tables.
   The new tracking class for the business document can also provide a MAINTAIN method which has as typed attributes as well as other methods which are not part of the interface /GBT/IF_TRC_OBJ.

   The Method /GBT/IF_TRC_OBJ~QUERY needs only be implemented if a search with Quick search is needed. The Query can also be used to implement a DDIC Search help.

   Hint: Use class /GBT/CL_GEN_BO to implement the new class will shorten the development time. Than only the data base tables need to be declared (constant instance attributes) to get maintain, save and read methods.

   CV_DB_HDR = 'ZGBT_O_<...>_HDR'
   CV_DB_ITM = 'ZGBT_O_<...>_ITM'

   The main structures used for the new business and a search help needs to populate with method /gbt/if_trc_obj~get_obj_info.

```
es_obj_info-key_struct_name      = 'ZGBT_S_<…>_KEY'.
es_obj_info-label_struct_name    = 'ZGBT_S_<…>_UI_LABEL'.
es_obj_info-overview_struct_name = 'ZGBT_S_<…>_UI_OVERVIEW'.
es_obj_info-node_struct_name     = 'ZGBT_S_<…>_NODE_STRUCT'.
es_obj_info-shlp_name            = 'ZGBT_<…>_TD_SH'.
```

   Hint: The Search help can be implemented by using the Query if using a search help exit by using the method /gbt/cl_trc_obj=>shlp_exit_select for selection.

4. Add a new object type to the customizing.
   Define Generic Object Types
   (Customizing Path: Cross-Application Components -> Processes and Tools for Enterprise Applications -> Settings for BO Framework and Navigation-> Define Generic Object Types)
   a. Define an Object type (starting with Z max. 10 characters)
   b. Use a description
   c. Enter the key structure 'ZGBT_S_<…>_KEY'
   d. Choose an Icon

   Define Objects for Global Batch Traceability Network in Customizing.
   Choose a 3 character Object type (starting with Z)

   (Customizing Path: Cross-Application Components -> SAP Global Batch Traceability -> Business Add-Ins (BAdIs) for SAP GBT -> Definition of Object Types -> Define Objects for Global Batch Traceability Network)
   a. Define if the Object is used as connector (like material document) or not (like Purchase Order)
   b. Enter the new class which has the Interface implemented/GBT/IF_TRC_OBJ
   c. Enter the Generic Object type (defined in the step before)
   d. Enter the descriptions

5. Activate new BAdI implementation in GBT for BAdI /GBT/EX_IL_LOAD_DETAIL
   This step is only needed if the Business documents details are derived from ERP.

   A BAdI implementation is needed as described below.

Also the new GBT Events may be needed and have to be added into ERP implementation.

If the business documents are standard ERP Objects this may be done in the appropriate ERP BAdI or otherwise added to the Object implementation.

3.2.3 UI extension

The UI is based on Floor Plan Mangager (FPM) for WebDynpro ABAP and Microsoft Silverlight for the Graphic. The Home Screen is build with the WebDynpro ABAP Page Builder and offers several search possibilities. Which elements are shown can be configured. For FPM see also FPM - Slides.

Extensions to the UI are Configuration of the Floor Plan Manger. Navigation also to other Systems is done with the Lauchpad. All Objects are displayed in the graphic out of the box.

3.2.3.1 Field Enhancement of UIs

3.2.3.1.1 Field enhancement for detail screens

The detailed screens are WebDynpro-screens which can be adjust by Floor Plan Manager Configuration Editor.

3.2.3.1.2 Add Column to TopDown- and/or BottomUp Table View

Precondition: The field, to be shown in the Tree View is available in the appropriate Node Structure (e.g. for Batch in Structure /GBT/S_BAT_NODE_STRUCT).

Depending on where the field should be displayed (Top-Down view (TD) and/or the Bottom-Up view (BU) the implementation is slight different.

- Add field to Display Structure

First step is to enhance the structure which contains the fields needed to display. If the field should be displayed in

1. For both, Top-Down and Bottom-Up view, create an append structure for structure /GBT/S_NW_TREE_DATA.
2. Only for the Top-Down view, create an append structure for structure /GBT/S_NW_TREE_TD.
3. Only for the Bottom-Up view, create an append structure for structure /GBT/S_NW_TREE_BU.

- Set field value in Display Structure

Create an implicit enhancement point for the appropriate method of class /GBT/CL_NW_READ and add the coding to set the field value. If you created the field in

4. /GBT/S_NW_TREE_DATA, choose method GET_NODE_DATA_TREE.
5. /GBT/S_NW_TREE_TD, choose method GET_NODE_DATA_TREE_TD.
6. /GBT/S_NW_TREE_BU, choose method GET_NODE_DATA_TREE_BU.

- Add field to Screen by Customizing within the Floor Plan Manager

Customize the screens with the table views and add the field to the Web Dynpro Component configuration. If the field should be displayed in

7. The Top-Down tree, add the field to /GBT/WDC_NW_TREE_TD_CFG.
8. The Bottom-Up tree, add the field to /GBT/WDC_NW_TREE_BU_CFG.

Create customizing of configuration /GBT/WDC_*_CFG for component.

1. Open NWBC and navigate to the screen you want to enhance.
2. Open technical help of workitem list.
3. Navigate to the ‘Component Configuration’ of ‘Current View’

4. Create configuration (part of Additional Function)
5. Select an transport.
6. scroll down to see the UiBB Schema
7. In a List UiBB you might want to add columns
   Important: For Screen which supports already save a field, which is editable, can be set to Display Type = 'Input Field' to enable editing.
   Additional coding is need (see here)!
8. Save your entries.
   Within the preview section, the columns are already visible.

3.2.3.2 Home Screen

Create Chips and add them with the WebDynpro ABAP Page Builder (ToDo: Example for Search/Quick Access)

3.2.3.3 UI Enhancements for Customer Objects

3.2.3.3.1 Introduction

GBT uses the Floor Plan Manager (FPM) for its WebDynpro ABAP UIs.

- A new pages has to be created for object details screen in the OVP configuration for new business objects which are defined in table /GBT/C_TRC_OBJ with their OBJ_TYPE.

  Dialog box with:

  - Page Type = 'Dialog Box'
  - Page Id = <OBJ_TYPE>

  The new page contains the new configuration for FPM_FORM_UIBB_GL2 which has to be added to the wiring model.

3.2.3.3.2 Step by Step example

To integrate a new object type (in this example **ZZZ**) into the GBT UI, the following enhancements have to be performed:

1. Create a new class which implements interface /GBT/IF_TRC_OBJ.
2. Create new GBT object type **ZZZ** using customizing activities "Define Generic Object Types" and "Define Objects for Global Batch Traceability Network".
3. Create a display structure ZGBT_S_**ZZZ**_NODE_STRUCT which contains fields to be displayed on the UI.
4. Create a Web Dynpro Configuration Z_WDC_**ZZZ**_DETAILS_CFG "Details of **ZZZ** in SAP GBT" for Web Dynpro Component FPM_FORM_UIBB_GL2 (Form GUIBB). Enter /GBT/CL_FDF_TRC_OBJ as the feeder class with parameters
   - Application Building Block ID: GBT
   - Node Name: **ZZZ**
   - RETRIEVE by Association: not set

5. In the Component Customizing create a dialog box for OVP /GBT/WDC_NW_OVP_CFG:

   On the tab 'Navigation' create a page with
   - Page Type: Dialog Box
   - Page ID: **ZZZ**
   - Title: <Header text for the Dialog Box>

   In the section 'Attributes of Page' set the following Standard Attributes:
   - Button Sets: Close
   - Width: 900
   - Height: 400

On the tab Overview Page Schema, create a section and add a UIBB of type 'FPM_FORM_UIBB_GL2' with

- Component: FPM_FORM_UIBB_GL2
- Window Name: FORM_WINDOW
- Configuration Name: Z_WDC_**ZZZ**_DETAILS_CFG

On the tab 'Wire Schema' add a row with:

- Component: FPM_FORM_UIBB_GL2
- Configuration Name: Z_WDC_**ZZZ**_DETAILS_CFG
- Source Component: /GBT/WDC_OBJ_NAV_KEY
- Port Type: Collection
- Port Identifier: OBJECT_DETAILS
- Connector Class: /PLMU/CL_FRW_W_CONN_DEFAULT

3.2.4 Extension of UI Navigation

GBT offers the functionality to navigate to the original system of the data record displayed. This is possible on every details screen within the UI of GBT.

Details screens can be customized by the customer to show additional fields or remove some of the fields displayed in standard GBT. In addition to this it is possible to customize navigation buttons on the details screens. For example a customer can add a button to the purchase order details screen to jump to the vendor master record in ERP. For the navigation Launchpads are used which needs to be customized.

The following describes the steps which need to be executed to add an additional navigation to another system.

1. Define Parameter for Parameter Mapping:
   Cross-Application Components -> Processes and Tools for Enterprise Applications -> Settings for BO Framework and Navigation -> Navigation, Parameter Mapping, and Services -> Define Parameters for Parameter Mapping (cross-client customizing)

   If the parameter you want to pass to the external system is not yet defined, you have to create a new entry in this table.

   Example:
   Define Parameter "VENDOR_NO" with description "Vendor Number" and assign data element "/GBT/VENDOR" to this parameter.

2. Define Mapping of Structure Components to Parameters:
   Cross-Application Components -> Processes and Tools for Enterprise Applications -> Settings for BO Framework and Navigation -> Navigation, Parameter Mapping, and Services -> Define Mapping of Structure Components to Parameters (cross-client customizing)

   In this customizing activity you define the mapping of the structure components used in the details screens to the parameters defined in the previous step. This enables passing of data from the details screen to the Launchpad parameters.

   Example:
   You create a new entry for field "VENDOR" of table name "/GBT/S_PURI_NODE_STRUCT", which is the table used for display of Purchase Order Item details in GBT. You map this field "VENDOR" to your newly defined parameter "VENDOR_NO".

3. Define Navigation Targets:
   Cross-Application Components -> Processes and Tools for Enterprise Applications -> Settings for BO Framework and Navigation -> Navigation, Parameter Mapping, and Services -> Define Navigation Targets (cross-client customizing)

To create an additional navigation functionality for a details screen you define a navigation target which will be used to determine the action you want to execute on the screen. Each Navigation Target will result in a separate FPM Event in the customizing of the details screens.

Example:
Create a new navigation target called “DISPLAY_VENDOR” with Description “Display Vendor Record”.

4. Create a new Launchpad application:
   Cross-Application Components -> SAP Global Batch Traceability -> Analysis of Global Batch Traceability Network -> Navigate to External Systems Using Launchpad

   A Launchpad application defines the application you want to include in your existing UI. This application can be on the same system but can also be any other system. There are several types of Launchpad applications you can define. Most popular are SAP Transaction and URL. Within the Launchpad application you define the parameters used for navigation and the target application itself.

   Example:
   Create a new Launchpad application by copying an existing application (for example “Display Batch in SAP ERP”). Enter a new name e.g. “Display Vendor in SAP ERP” and change the transaction code to ME13.
   Set the Application Alias to a new value e.g. “LPD_VEND_DISP”. This value will be used in the next step to link this Launchpad application to a navigation target.
   In section Target App. Parameters you have to define the mapping of parameters to the target transaction parameters. Click on the Parameter Mapping Button and maintain the following:

Program:	SAPMM06I
Dynpro Number:	0100
OK Code:	<space>
Forward only the following parameters:	<checked>
Not completed by COMMIT:	<checked>

   In the table for parameters enter the following two parameters and their target parameters:

Source Parameter:	VENDOR	Target Parameter:	EINA-LIFNR
Source Parameter:	MATNR	Target Parameter:	EINA-MATNR

   Save the Launchpad application.

5. Assign the Navigation Target:
   Cross-Application Components -> Processes and Tools for Enterprise Applications -> Settings for BO Framework and Navigation -> Navigation, Parameter Mapping, and Services -> Assign the Navigation Target

   In this step you create the link of the internal GBT data and navigation structure to the external navigation using Launchpad technology. For each object type, application building block and navigation target combination you define the link to the Launchpad role, instance and application alias you want to use.

   Example:
   Create a new entry in this table with the following data:

Object type:	GBT_PURI
ABB ID:	GBT
Nav. Target:	DISPLAY_VENDOR
Role:	GBT
Instance:	GBT_SAP_EXT_SYS
App. Alias:	LPD_ VEND_DISP

6. Adjust the Detailed Screen as described above.
   Example:
   Create a customizing for the purchase order details screen. In this customizing add a new button or link and assign the FPM Event “DISPLAY_VENDOR” to the button or link.

3.2.5 Extent load of Business Documents Details

To fill the extended data (additional fields or even new Objects) in GBT the integration needs to be adapted.

3.2.5.1 Integration of additional attributes from ERP

The following steps are needed:

1. Create new RFC Structures in ERP and GBT
2. Create selection class in ERP (e.g. by copy and adaption of existing class)
3. Create BAdI implementation in GBT (e.g. by copy and adaption of existing implementation)
4. Activate new BAdI implementation in GBT

3.2.5.1.1 Extension in ERP

(1) Create new RFC Structure
Example: copy /GBTINT/S_PURI_DATA_RFC to ZGBTINT_S_PURI_DATA_RFC and exchange the table type of component OBJ_HDR and/or OBJ_ITM to the table types which has the extended structures.

(2) Create a new selection class with interface /GBTINT/IF_OBJ_ACCESS
Example: copy /GBTINT/CL_PURI_ACCESS to ZGBTINT_CL_PURI_ACCESS

a. fill the new created RFC structure in method /GBTINT/IF_OBJ_ACCESS~GET_DETAILS
Example: exchange type /GBTINT/S_PURI_DATA_RFC and /GBTINT/S_PURI_HDR_RFC and/or /GBTINT/S_PURI_ITM_RFC with the extended ZGBTINT -Type and adjust the selection if needed.
Usually only change the type of the variable which has the different RFC-Structure is needed.

b. populate the used structures with method /GBTINT/IF_OBJ_ACCESS~ GET_INFO
Example:

```
es_info-key_struct       = '/GBTINT/S_PURI_KEY_RFC'.
es_info-data_list_struct = 'ZGBTINT_S_PURI_DATA_RFC'.
```

3.2.5.1.2 Extension in GBT

(1) Create new RFC Structure
Example: copy /GBT/S_PURI_DATA_RFC to ZGBT_S_PURI_DATA_RFC and exchange the table type of component OBJ_HDR and/or OBJ_ITM to the table types which has the extended structures.

(2) Implement BAdI /GBT/EX_IL_LOAD_DETAIL
(Customizing Path: Cross-Application Components -> SAP Global Batch Traceability -> Business Add-Ins (BAdIs) for SAP GBT -> Integration BAdI: Loading of Object Details from External System)

a. Create a new BAdI implementation for BAdI /GBT/EX_IL_LOAD_DETAIL
Example: copy /GBT/CL_EX_IL_LOAD_DETAIL_PUI to ZGBT_CL_EX_IL_LOAD_DETAIL_PUI

fill the new created RFC structure in GBT in method /GBT/IF_EX_IL_LOAD_DETAIL~LOAD_DETAIL by calling the new created ERP selection class with function module '/GBTINT/RFC_LOAD_DETAIL' and use the ABAP command 'IMPORT' to fill the new created GBT RFC structure.

Example:

```
DATA lv_data TYPE /gbt/xstring.

CALL FUNCTION '/GBTINT/RFC_LOAD_DETAIL'
  DESTINATION iv_rfc_dest
  EXPORTING
    iv_obj_type    = iv_erp_obj_type
    iv_keys        = lv_keys
    iv_class_name = 'ZGBTINT_CL_PURI_ACCESS'
```

```
    IMPORTING
      ev_data       = lv_data.

* uncompress data
  DATA ls_obj_data_rfc TYPE zgbt_s_puri_data_rfc.

    IMPORT object_data = ls_obj_data_rfc " <= GBTRINT-SP10
           obj_hdr     = ls_obj_data_rfc-obj_hdr  ">= SP11
           obj_itm     = ls_obj_data_rfc-obj_itm  ">= SP11
           FROM DATA BUFFER lv_data
      ACCEPTING PADDING
      ACCEPTING TRUNCATION.
    " <-RFC
```

Activate BAdI in GBT

i. Check the filter values of the new BAdI Implementation
   Example: PUI = OBJ_TYPE
ii. deactivate the standard GBT BAdI implementation
iii. activate the new BAdI implementation

Remark: instead of using the ABAP-command IMPORT FROM DATA BUFFER it is also possible to use method `/gbt/cl_exim_util=>import_data`. This method has the advantage also to accept different field length as well as extended structures.

3.2.5.2 Integration of new business documents

This is mainly expected for non-SAP ERP systems. But also this may be needed for customer specific business documents in ERP.

The kind of integration needs to be provided by the implementation project. It depends if a PUSH scenario (like used by a File Import) or a PULL scenario (like used for transferring GBT-Events from ERP to GBT) can be chosen.

3.3 Extension of missing events

3.3.1 Field extension

You can add customer specific fields to table /GBT/N_MS_EV_OBJ and add these fields as search criteria in the missing events UI.

Add customer specific fields to table /GBT/N_MS_EV_OBJ and add these fields as search criteria in the missing events UI doing the following steps (see SAP Note 1889447):

1. Create the customer include CI_OBJ_MS_EVNT_ATTR if necessary.
Define your fields in the customer include.

2. Copy standard search help /GBT/OBJ_MISS_EVENT_SH to create your customer search help with the new fields defined in the customer include.
NOTE: Make sure to use the same selection method /GBT/N_MS_EV_OBJ in your customer search help.

3. Add your customer search help to the standard search help by EITHER

a) Creating an implementation for BAdI definition /PLMB/EX_SPI_METADATA to add your customer search help as query.
See the correction instruction for an example coding.

OR

b) Appending your customer specific search help to the standard collective search help /GBT/OBJ_MISS_EVENT_CSH.

4. Create a customizing for the missing events search GUIBB and enter your customer search help in feeder class parameter (Button 'Feeder Class Parameters' in section General Settings).

5. In the customizing, add the new fields from customer include CI_OBJ_MS_EVNT_ATTR as selection parameters to the missing events search GUIBB.

6. If you want to display the new fields also in the result list for missing events, create a customizing for the result list GUIB and add the new fields from customer include CI_OBJ_MS_EVNT_ATTR to the result list GUIBB.

3.3.2 Configuration of Mandatory Fields

Manual added Events via “Missing Event” functionality it may require mandatory information like e.g. “Reason Code”. To configure a field as mandatory you have to change the Web Dynpro Configuration in Administration Mode. You have to navigate to the screen where the field is shown. There, with a right click on the field, you can change the “Settings for Current Configuration”. Switch the state of the chosen field to “Required Entry” and save the changes.

3.4 Extension of Worklist

In most cases it is expected that the Worklist Item needs to be extended. But also Workgroup and Worklist header can be enhanced.

3.4.1 Workgroup

Append own fields to workgroups by creating the customer include **CI_WL_GROUP_ATTR**.

Add the new fields to the UI by creating customizing for:

- Workgroup List: List UIBB configuration /GBT/WDC_WL_GROUP_LIST_CFG.
- Workgroup Details: Form UIBB configuration /GBT/WDC_WL_GROUP_CFG.

3.4.2 Worklist Header

Append own fields to worklists by creating the customer include **CI_WRKLST_HDR_ATTR**.

Add the new fields to the UI by creating customizing for:

- Worklist List: List UIBB configuration /GBT/WDC_WORKLIST_LIST_CFG.
- Worklist Details: Form UIBB configuration /GBT/WDC_WORKLIST_CFG
- Create Worklist: Form UIBB configuration /GBT/WDC_WORKLIST_CREATE_CFG

3.4.3 Worklist Items

Append own fields to worklist items by creating the customer include **CI_WRKLST_ITM_ATTR**.

Add the new fields to the UI by creating customizing for:

- Worklist Item List: List UIBB configuration /GBT/WDC_WORKLIST_ITEM_LIST_CFG.

3.4.3.1 Example: add an editable field to Worklist Items

Set a new field to editable is possible by creating an enhancement in method /PLMB/IF_SPI_PROPERTIES_ACCESS~GET_PROPERTIES in class /GBT/CL_NW_SP.

The following example coding shows how to set the field ‘NOTE’ in the worklist item to editable for already existing entries (CHANGE).

```
  DATA:
    lt_worklist_item_key TYPE /gbt/t_worklist_item_key.


  FIELD-SYMBOLS:
    <ls_worklist_item_key> LIKE LINE OF lt_worklist_item_key.


  CASE iv_reference_type.
*--------------------------------------------------------------------*
    WHEN /plmb/if_spi_c=>gs_c_prpty_ref_type-data_record.
      CASE iv_node_name.
        WHEN /gbt/cl_nw_mp=>gc_node_name-worklist_item.
*         No changes for aggregated nodes!
          lt_worklist_item_key = ig_reference_data.
          LOOP AT lt_worklist_item_key ASSIGNING <ls_worklist_item_key>.
            CLEAR ls_property_index.
            CLEAR ls_indexed_prp_multi.
            ls_property_index-property_index = sy-tabix.
            INSERT ls_property_index
                   INTO TABLE ls_indexed_prp_multi-property_index.
            IF /gbt/cl_nw_eng=>node_is_cumulation_node(
                      <ls_worklist_item_key>-obj_guid ) = abap_false.
              ls_property-option = /plmb/if_spi_c=>gs_c_property-changeable.
            ELSE.
              ls_property-option = /plmb/if_spi_c=>gs_c_property-readonly.
            ENDIF.
            ls_property-node_field = 'NOTE'.
            INSERT ls_property INTO TABLE ls_indexed_prp_multi-properties.
            APPEND ls_indexed_prp_multi TO et_properties_multi_idx.
          ENDLOOP.
      ENDCASE.

  ENDCASE.
```

Automatically trigger a SAVE-event after changing an editable field by creating an enhancement in method /PLMU/IF_FRW_G_ACTIONS~PROCESS_ACTION_EVENT in class /GBT/CL_FDL_WORKLIST_ITEM.

The following example coding shows how to automatically trigger a SAVE-event after changing the content of field ‘NOTE’ in the worklist item.

```
  CASE io_event->mv_event_id.

    WHEN if_fpm_guibb_list=>gc_guibb_list_on_cell_action.
      io_event->mo_event_data->get_value(
        EXPORTING
          iv_key   = if_fpm_guibb_constants=>gc_guibb_attributes-field_name
        IMPORTING
          ev_value = lv_field_name ).
      CASE lv_field_name.
        WHEN 'NOTE'.
*         Direct SAVE
          lo_event = cl_fpm_event=>create_by_id(
              iv_event_id = if_fpm_constants=>gc_event-save ).
          cl_fpm_factory=>get_instance( )->raise_event( lo_event ).

*         Switch back to edit mode
          lo_event = cl_fpm_event=>create_by_id(
              iv_event_id = if_fpm_constants=>gc_event-edit ).
          cl_fpm_factory=>get_instance( )->raise_event( lo_event ).

      ENDCASE.
  ENDCASE.
```

4 Master data

Master data in GBT are optional and not mandatory for GBT. Nevertheless it is useful to have them to get search possibilities, text, alternative IDs and some common attributes. Master data are integrated from ERP with ALE / IDOC.

If additional attributes needed e.g. for reporting in GBT the IDOC-Inbound logic needs to be extended. For this kind of extension the appropriate GBT-Inbound Function module (/GBT/IL_IDOC_IN_*) should be replaced by a new one (a modified Copy of the GBT standard IDOC function module) which has the needed attribute value mapping into the target Object.

Remark: UI for Master data is SAP-GUI.

4.1 Material

The ERP Material is stored within the Product (based on the implementation used in CRM and SRM) (AP-MD-PRO) with type material (01). The integration is specific to GBT and includes the mapping between ERP Material and GBT Product Key.

Attributes taken over by the integration are material description, unit of measures (UoM), Global Trade Item Number (GTIN).

4.1.1 Product Category to group Materials

Material attributes used for grouping and needed in GBT for search of materials. This can be e.g. the material type (MARA-MTART) or any other MARA attribute as well as Z-fields if they are part of the Material IDOC Segment for MARA. It is also possible to use plant specific fields like production planner (MARC-DISPO).

4.1.1.1 Configure ERP Material attribute as Product Category

To use one of the Material attributes as Product Category in GBT, use report /GBT/R_PROD_CAT_SETUP (for details check the documentation). The attribute will be automatically added to the Material Search Help.

If this Product Category should also be directly used in the Batch Search Screen the Product Category needs to be appended to structure /GBT/S_BAT_NODE_STRUCT. The data element is defined in class /GBT/CL_PROD_GROUP_ERP_<attribute name> => GV_ROLLNAME.

Attention: Don't add the requested attribute to the CI-Includes. This would also extend the Database.

Existing Classes for Product Category Integration:

Class Name	ERP Table	Description
/GBT/CL_PROD_GROUP_ERP_MATKL	MARA	Retrieve Material Type from ERP as Prod. Cat. Hierarchy
/GBT/CL_PROD_GROUP_ERP_MTART	MARA	Retrieve Material Type from ERP as Prod. Cat. Hierarchy
/GBT/CL_PROD_GROUP_ERP_PRDHA	MARA	Retrieve Product Hierarchy from ERP as Prod. Cat. Hierarchy
/GBT/CL_PROD_GROUP_ERP_SPART	MARA	Retrieve Division from ERP as Prod. Cat. Hierarchy
/GBT/CL_PROD_GROUP_ERP_MAGRV	MARA	Retrieve Packaging MatlGrp from ERP as Prod. Cat. Hier.
/GBT/CL_PROD_GROUP_ERP_DISPO	MARC	Retrieve MRP Planner from ERP as Prod. Cat. Hierarchy

/GBT/CL_PROD_GROUP_ERP_FEVOR	MARC	Retrieve Prodn. Supervisor from ERP as Prod. Cat. Hierarchy

Create a new Class for Product Category Integration

If an another attribute from MARA, which is used to group Material, should be used as Product Category the interface /GBT/IF_PROD_GROUP needs to be implemented for this.

1. Global Definitions

Global value	Description
GV_FIELDNAME	Fieldname used in ERP MARA or MARC
GV_ROLLNAME	Date element used in GBT for this Field
GV_TABKIND	'A' for MARA or 'C' for MARC Constant values CV_MARA, CV_MARC defined
GV_EX_SEGNAM	Used for customer defined MARA field extensions which are transferred in an Customer defined IDOC-Segment (not Supported for MARC-fields)

2. Methods which needs to be implemented

Method Name	Description
READ_AS_CATEGORY	Read Product Group values from external Source as Category. This method does the extraction of the possible values for the specified attribute
MAP_TO_CATEGORY_ID	Map Product Group value to category ID This method defines the value representation in GBT.

Tip: Check existing Implementations of this interface (e.g. /GBT/CL_PROD_GROUP_ERP_MTART)

4.1.2 Extension of Product

The Product can be extended by creating a new Set type (Transaction COMM_ATTRSET). New set types can be filled by the ALE/IDOC Integration (BAdI-Method /GBT/EX_IL_MATMAS MAP_TO_PRODUCT (Note 1828085 / SP03). For Set type see documentation:
http://help.sap.com/saphelp_crm50/helpdata/en/91/d8fc377cb5be5ae10000009b38f842/frameset.htm
http://help.sap.com/saphelp_crm70/helpdata/en/46/5786ec01a208e7e10000000a114a6b/content.htm?frameset=/en/46/57672501a208e7e10000000a114a6b/frameset.htm

For an extension of ALE / IDOC for inbound processing in GBT see here.

4.1.2.1 How to create a set type

1. Call Transaction COMM_ATTRSET (Development authorization is needed)
2. Create all needed Attributes (using COMM_ATTRSET)
3. Create the Set type (also using COMM_ATTRSET) for Product Extension

Definition:

Set Product Type to 'Material'

Org. Dependency is independent

Don't check for "multiple use"

Assign the new attributes you need to extend the product with.

Please check:

1. A Table with the name of the Set Type you choose should exist.
   Remark: the fieldname of the attributes in the table are generated.
2. Function modules: ZOM_<table name> * should exists

4. Transport Set Type
   Use Transaction COMM_ATTRSET Start Menu: Set Types/Attributes -> Transport -> Set type

   Please check:
   1. Structure COMT_PROD_MAT_MAINTAIN_API must have a new Append structure with an internal table for the created 'SetType'.

5. Assign Set Type to Main Category (e.g. GBT_MATERIAL)
   a. Call Transaction COMM_HIERARCHY (or via SPRO => Cross Application Components -> SAP Product -> Product without Customizing Transfer ..=> Maintain Categories/Hierarchies)

      1. Chose Tab 'Find' Select Category (e.g. GBT_MATERIAL)
      2. Chose Tab 'Set Types' add your created Set type to these Category.

4.2 Business Partner

The ERP Business Object Plant, Customer and Vendor are stored as Business Partner (AP-MD-BP) in GBT. The integration is specific to GBT and includes the mapping between ERP and GBT Business Partner Key.

The standard address with its attributes and Global Location Number are transferred by the integration.

4.2.1 Extension of Business Partner

The Business Partner can be extended by using the Easy Enhancement Workbench (EEW). Additional attributes which should be filled by the ALE/IDOC Integration must be supported by a new IDOC Function Module.

For an extension of ALE / IDOC for inbound processing in GBT see here.

4.3 Descriptions

Codes to identify a Category, a Type or an Object often have an additional textual description. Codes and their descriptions may be specific to the sending system (e.g. by customizing).

Attention: Descriptions for attribute values which are used to group materials are handled by Product Category within GBT.

To load descriptions form ERP into GBT the use of report /GBT/R_DESC_TABLE_SETUP (for details check the documentation) is recommended. New kinds of description for codes can be added by implementing the interface /GBT/IF_DESC_LOAD.

Existing Classes which implement the interface:

Class Name	ERP Table	Description
/GBT/CL_DESC_LOAD_ERP_T001L	T001L	Retrieve Name for Storage Location (not language dependent)

/GBT/CL_DESC_LOAD_ERP_T156T	T156T	Retrieve Material Movement Description (Standard)
/GBT/CL_DESC_LOAD_ERP_T156T900	T156T	Retrieve Material Movement Description (Custom specific)
/GBT/CL_DESC_LOAD_ERP_T163Y	T163Y	Retrieve Description for Item Categories used in Purchasing Documents

4.3.1 Add a new Class for Description Integration

If descriptions of a specific table be integrated from to GBT the interface /GBT/IF_DESC_LOAD needs to be implemented.

1. Global Definitions

Global value	Description
GV_TABNAME	Table used as Identifier for set of descriptions
GV_DESCRIPTION	Text which identifies for which type of values the descriptions are used (value can be set in the class constructor depending on login language)
GV_LOCAL_SCOPE	Expected scope global => used in all Systems local => used for one System
GV_SWITCH_SCOPE	Allow to switch the scope

2. Methods which needs to be implemented

Method Name	Description
READ	Read Description from external Source and store to buffer. This method does the extraction of the possible values for the specified attribute
SAVE	This method stores the buffered values to data base.
CLEAR	Clear buffer

Addition: To see the descriptions in the UI extend /GBT/S_*_NODE_STRUCT (but not the CI-Includes!) and extend the implementation of /GBT/IF_TRC_OBJ->GET_UI_DATA_BY_GUID.

4.3.2 Store Description to database table /GBT/D_CODE_GEN

The descriptions can be stored in the standard GBT database table /GBT/D_CODE_GEN which offers a central place for various sets of descriptions. These sets are separated by the name of the different data elements which are used for the different code-sets.

This table can be used if the codes, for which a description is needed, can be stored with 4 Characters and the description is fully specified by Code and Language.

Attention: This table cannot be used if the key needs more than 4 Characters or is not described by a single code. This table should also not be used if a big set of descriptions is expected.

Class /GBT/CL_CODE_GEN_DESCR is used to maintain and read the description form the central table /GBT/D_CODE_GEN.

As example how to deal with this class check implementation of /GBT/CL_DESC_LOAD_ERP_T163Y and also method ENRICH of class /GBT/CL_TRC_BO_PURI.

4.3.2.1 Search help for Code values based on table /GBT/D_CODE_GEN

A generic Function module /GBT/CODE_GEN_SHLP_EXIT for the search help exit offers the search within the table /GBT/D_CODE_GEN.

DTEL is a key of table /GBT/D_CODE_GEN which groups a set of code.

The search help which uses the generic FM needs to set the attribute DTEL with the name of the corresponding data element as default value.

As example see data element /GBT/PURO_PSTYP and search help /GBT/PURO_PSTYP_SH.

How to extend the UI with additional description fields see chapter UI extension.

5 Extension of ALE / IDOC

If an extension of the ERP IDOC type is need please refer to SAP- Documentation of Extending an IDOC type.

Attention: Customer specific IDOC-Segments-Types need a name like **'Z1*'**, e.g. a name has to start with 'Z1' like in 'Z1SEGMENT_TYPE'.

5.1 Configure use of an IDOC Inbound Function Module in GBT

The following steps are needed to use a new created IDOC Function Module in GBT:

1. Transaction BD51 - Maintain Attributes of the Inbound Function Modules
2. Transaction WE57 - Assign Inbound Function Module to Message Types and IDOC Types
3. Transaction WE42 - Define Process Codes

To configure the ALE/IDOC on GBT check also Basic Settings for SAP GBT 2.0.

6 Extent Filter and Aggregation Possibilities by custom Simplifications and Views

6.1 Introduction

Beside the predefined Network Views you can define own views using the predefined simplifications. Furthermore, own simplifications can be defined.

- For more information about the delivered views, see Customizing for SAP Global Batch Traceability under *Analysis of Global Batch Traceability Network -> Analysis of Global Batch Traceability Network -> Define Views for Filtering and Aggregation*.

- For more information about the delivered simplifications, see the transaction for view cluster maintenance (transaction SM34) for view cluster /GBT/VC_SIMP.

Note

A view summarizes several simplifications. A simplification reduces the complexity of a global batch traceability network by either filtering nodes out of the network, or by aggregating many nodes into a few aggregation nodes. A simplification defines a specific selection of network nodes to which it is applied. If the simplification is an aggregation, it also contains the aggregation key fields. This enables you to combine multiple simplifications.

On the user interface, you select such a combination simply by selecting a view.

On the user interface, the filtered network nodes or events are deleted and not displayed in a network. In comparison, aggregated network nodes or events are replaced, and displayed as aggregated network nodes or events.

Examples

- You can define own simplifications like the following:
  - To filter all nodes but purchase orders and batches
  - To filter all nodes but production orders and batches
  - To cut segments at a certain purchase order
  - To cut segments bottom up at a certain material

- You can define own views like the following:
  - Batch Only on Plant Material Level
  - Outbound Delivery by Ship-To Party

6.2 Define own Simplifications for Filtering

1. Start the transaction for the view cluster maintenance (transaction SM34).

2. Enter /GBT/VC_SIMP as view cluster name and *Maintain*.

3. For the simplification maintain the following:

   a. Define a simplification ID.
   b. Choose simplification type *Filter*.
   c. Choose the simplification object that is on which object the simplification makes an impact: on a network node or on a relationship.
   d. Enter a description of the simplification ID

4. In the *Dialog Structure* choose *Selection of Nodes or Events*.

5. Maintain the following:

   a. Define a selection ID.
   b. Choose a selection criterion.
      Enter the field name of an object attribute as it is defined in the corresponding dictionary table /GBT/S_<**table identification**> _NODE_STRUCT.

   Example

   - If you want to create a selection for batches using a certain plant, enter **PLANT** as the selection criterion.
     You find this field name in table /GBT/S_BAT_NODE_STRUCT.
   - If you want to create a selection for batches using a certain storage location, enter **STGE_LOC** as the selection criterion.
     You find this field name in table /GBT/S_BAT_NODE_STRUCT.
   - If you want to define a selection for the object type, enter **OBJ_TYPE** as the selection criterion.

   c. Define the selection option, for example, *Equals*.
   d. Include or exclude the selected objects from the network.

   Note
   For filtering, the combination of selection criterion *Equals* and selection option *Include in Selection* results in a **deletion** of the respective network nodes. They are no longer displayed in the network.

   e. Define the data type, for example, *Character*, of the selected objects.
   f. Define the value of the selection criterion, for example, *BAT* for objects of type batch or *DII* for objects of type inbound delivery.
   g. Save your entries.

   Note
   For filtering, you have nothing to define in the *Dialog Structure* under *Definition of Aggregation*.

6.3 Define own Simplifications for Aggregating

1. For step 1, 2, 3, and 4, see chapter *Define own Simplifications for Filtering*.
   In step 3, choose simplification type *Aggregation* instead of simplification type *Filter*.

2. For step 5, see chapter *Define own Simplifications for Filtering.*

   Note
   For aggregating, the combination of selection criterion *Equals* and selection option *Include in Selection* results in a **selection** of the respective network nodes. They are the input for the aggregation in step 3.

3. In the *Dialog Structure* choose *Definition of Aggregation*.

4. Maintain the following:

a. In column *Property ID* enter one or more fields which are used to aggregate network nodes in one network node.

Example

If you want to define an aggregation of batches on plant level, you first select batches as nodes and second enter *BATNR, LOGSYS, MATNR*, and *PLANT* as property IDs. All nodes of object type BAT (batch) which have the same batch number, logical system, material number, and plant are aggregated and displayed as one node in the network.

Recommendation
If you want to support the usage of an aggregated network node as a root node, we recommend to **index the respective database tables** with prefix /GBT/O*, for example, /GBT/O_BAT_ITM. Otherwise, **performance issues** could occur.

6.4 Define Views for Filtering and Aggregation

1. In Customizing for SAP Global Batch Traceability (transaction SPRO) choose *Analysis of Global Batch Traceability Network -> Analysis of Global Batch Traceability Network -> Define Views for Filtering and Aggregation*.

2. Define a view ID and a text. This text is used on the user interface to select a certain view, and is added to the screen title to inform the user about the selected view.

3. Select one or more of the predefined simplification IDs that apply to your prerequisites. Depending on the simplification type, the network nodes or events are filtered or replaced with aggregated nodes or events on the user interface.

4. Save your entries.

Example to filter both all network nodes except batches and all relationships with a relationship quantity of zero (combination of views *BATCH_ONLY* and *REL_GT_ZERO_ONLY*)

If you select this view on the user interface, the system displays only batches and only relationships with a quantity that is greater than zero. All other nodes and relationships are filtered.

1. In this Customizing activity, under *Filter and Aggregation View*, enter **Z_BATCHES_AND_REL_GT_ZERO_ONLY** as the view ID and **Only Batches / Relations. > Zero** as the view text.

2. Under *Simplification*, select *FILTER_ALL_NODES_BUT_BATCHES* and *FILTER_REL_QUANTITY_EQ_ZERO* as the simplification IDs and save your entry.

3. On the user interface, you can choose the new view *Z_BATCHES_AND_REL_GT_ZERO_ONLY*.

6.5 Define a Default Network View

The Set/Get parameter ID /GBT/NETWORK_VIEW needs to be predefined for a user if a network view should be set as the new default view. Assign the view-ID (as defined in table /GBT/C_VIEW) as value to the parameter /GBT/NETWORK_VIEW with transaction SU3 (own data), SU01 or SU10 (mass maintenance).

7 Enable custom Use Cases with Reports and Worklists

Additional functionality can be added to GBT integrated into the Batch Search UI and also with the Worklist UI.

Special Reports, which are able to accept a list of GBT-Objects as input (see details below) can be developed. As example please check usage list, distribution list or others provided in GBT.

A complex selection of Objects, Batches or others, can also be programmed as dynamical Worklist. E.g. Batches which will soon reach their best-before-date and still on stock.

For the Worklist it is also possible to develop and add Actions like e.g. create a Quality-Notification or to block a Batch in ERP.

See also Extension of UI Navigation and Enhancement of Network Graphic.

7.1 Custom reports in GBT UIs

Reports can be directly included in the GBT user interface for the Batch Search and to the Worklist.

These reports need the specific Parameter p_handle, which allows the transfer from the Selected Data from the WebDynpro UI.

Also existing reports developed by SAP can be replaced or removed from the user interface. For the integration of reports in the GBT user interface the SAP NetWeaver functionality called Launchpad is used.

1. Create report

A new report has to be created in customer namespace. To create a new report, open the ABAP Editor (SE38) from the SAP Easy Access Menu Tools → ABAP Workbench → Development → ABAP Editor.

Enter the name of the new report in field Program and press the button Create.

In the popup enter the title of your report and select “Executable program” as type for your report.

The report must provide input parameter p_handle of type memory_id. This parameter receives a handle which points to a database table. The database table is filled with object ids from the UI when the user selects objects and requests to execute the report.

Example code:

```
PARAMETERS:  p_handle TYPE memory_id MEMORY ID handle.
To read the object ids from the database, implement the following data definitions and method calls:
DATA:
  lv_handle             TYPE        sysuuid_x16,
  lo_buffer_handle      TYPE REF TO /plmb/cl_frw_buffer,
  lt_parameter_set      TYPE        /plmb/t_frw_parameter_set.
* Get handle to read IDs from shared memory using framework functionality
/plmb/cl_frw_buffer=>get_buffer_handle(
  EXPORTING
    iv_lifetime =   3600   " Lifetime in seconds
  IMPORTING
    eo_handle   =   lo_buffer_handle   " Buffer class
).

lv_handle = p_handle.

* Read IDs from buffer
lo_buffer_handle->get_data(
```

```
  EXPORTING
    iv_handle       = lv_handle     " Object Type
    iv_clear_buffer = abap_true     " Supplement for True Boolean Type: 'X
' == True '' == False
  IMPORTING
    et_data         = lt_parameter_set
).
```

Save and activate your report.

2. Create a transaction

For the new report a transaction of type “report transaction” has to be created.

On the SAP Easy Access Menu choose Tools → ABAP Workbench → Development → Other Tools → Transactions.

Enter the name of the new transaction in field “Transaction code” and press button “Create”.

In the popup enter a short text for your transaction and select start object “Program and selection screen (report transaction)”. Press enter.

In the next screen enter your report name in field “Program”. Enter the screen number which includes the input parameter fields in field “Screen number”. In input area “Classification” set the flag for “SAPGUI for HTML”.

Save your transaction and test if the right report is called.

3. Configure the user interface to call the new report

All reports which can be called directly from the GBT user interface are registered within a launchpad. The launchpad is of role “GBT” and the instance is called “GBT_REPORTS”.

To include a custom developed report into the GBT user interface the SAP delivered GBT launchpad has to be changed. In the customizing for SAP Global Batch Traceability select User Interface Settings → Launchpad for integration of reports. The screen will show all linked applications in the left frame.

Add your report by clicking the button “New Application” and enter the following values:

Field name	Value
Link Text	<Display name of your report>
Application Type	Transaction
Transaction Code	<Transaction code you created before>
System Alias	SAP_LocalSystem
Force local system if NWBC	<checked>

Addition parameters:

Field name	Value
Application Alias	LPD_REPORT_<Add a abbreviation for the report>
Target App. Parameters	Click on the button for “Parameter Mapping”. In the area for batch input enter the name of your report and the dynpro number of the selection screen (default 1000). Check the option for “Forward only the following parameters”.

	In the area for additional options for call transaction activate checkboxes for “Not completed by COMMIT” and “No batch input for CALL trans.”. Select processing mode “Display errors” and update mode “Asynchronous”. Add one line to the table at the bottom of the popup and enter source parameter “HANDLE_ID” and target parameter “P_HANDLE”.
GUI type	SAP GUI for HTML
Entries once started	Skip initial screen if possible
Navigation mode	Headerless Portal Window
History mode	Navigation entry can occur once in history
Parameter Forwarding	Post parameters

Save the launchpad.

Now you should be able to access your report in the SAP GBT user interface.

Note: the first time you change a SAP delivered launchpad a new version of the launchpad will be generated and this new version will be used from there on. To access the launchpad originally delivered by SAP choose Extras → Show SAP Version.

For more details about changing an existing launchpad, read the documentation for launchpad.

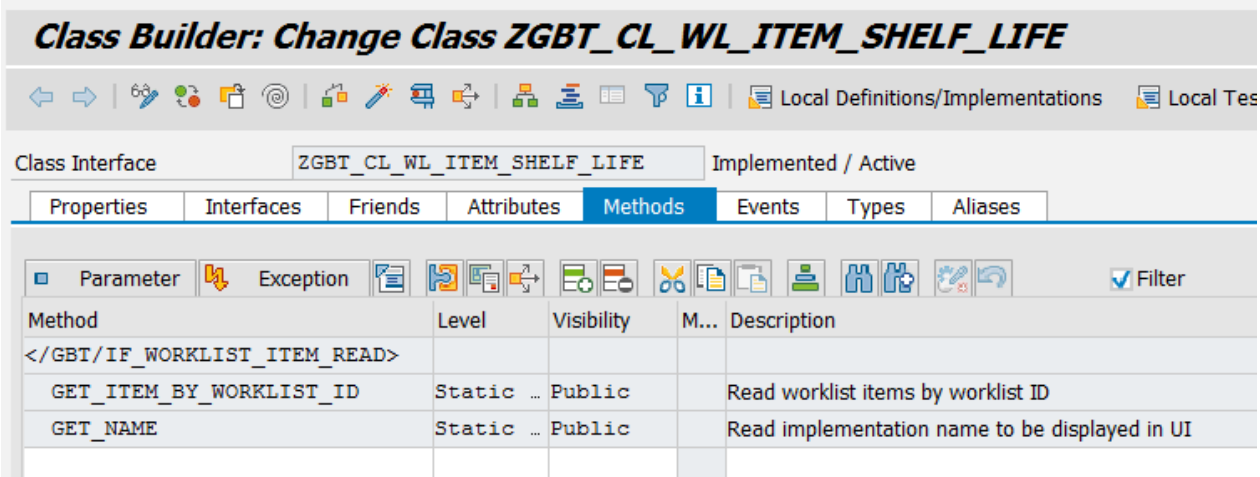
7.2 Dynamical determined Worklist Items

Worklist items can either be read from the standard database tables or they can be selected dynamically when displaying a worklist by executing a specific programmed determination.

Interface /GBT/IF_WORKLIST_ITEM_READ, which has two methods, needs to be implemented. Define a name for the dynamical determination of worklist items with GET_NAME and GET_ITEM_BY_WORKLIST_ID which has the worklist item determination logic.

In this example, a worklist is implemented to displays all batches which already expired until today. This worklist should always display the current result for the date it is displayed.

The implementation of interface /GBT/IF_WORKLIST_ITEM_READ has to be created using transaction SE24.



1. Double click on method *GET_NAME*, copy the following coding into this method, and choose

Save (Ctrl+S):

```
rv_name = 'Batches expired today'(001).
```

2. Go back to the class header (F3), and double click on method *GET_ITEM_BY_WORKLIST_ID*.
3. Copy the following coding into this method and choose Save (Ctrl+S):

```
DATA:
    lt_bat_itm_guid TYPE /gbt/t_obj_guid.

  FIELD-SYMBOLS:
    <ls_worklist_itm_key> LIKE LINE OF et_worklist_itm_key,
    <lv_bat_itm_guid>     LIKE LINE OF lt_bat_itm_guid.


  CLEAR et_worklist_itm_key.

* Perform selection to return items with real batches
  SELECT obj_guid FROM /gbt/v_bat_itm INTO TABLE lt_bat_itm_guid
    WHERE expiry_date <> '00000000'
      AND expiry_date < sy-datum.

  LOOP AT lt_bat_itm_guid ASSIGNING <lv_bat_itm_guid>.
    APPEND INITIAL LINE TO et_worklist_itm_key
                ASSIGNING <ls_worklist_itm_key>.
    <ls_worklist_itm_key>-worklist_id = iv_worklist_id.
    <ls_worklist_itm_key>-obj_guid    = <lv_bat_itm_guid>.
    <ls_worklist_itm_key>-obj_type    = 'BAT'.
  ENDLOOP.
```

4. Activate all objects of the new class (Ctrl+F3).
5. Add the new class to the GBT customizing *Analysis of Global Batch Traceability Network* → *Maintain Available Classes for Dynamic Worklist Items*.
6. Create a worklist with dynamic items:
   1. In the SAP NetWeaver Business Client (transaction NWBC) select a role and navigate to the *Process Worklists* screen.
   2. Enter a name and a description for the new dynamic worklist to show all batches expired today.
   3. From the dropdown menu for column *Item Selection* choose the class name created before ('Batches expired today').
   4. Select the new worklist and save it.
   5. Select the new worklist and choose *Display*.

7.3 Creating a (static) worklist and adding items via report

Creating a worklist and adding items is possible not only interactively in the UI but also via a report.

All methods can be found in class /GBT/CL_WORKLIST.
In general, the following steps have to be performed:

1. On the entry screen, a worklist ID and description are entered plus some selection criteria if needed.
2. Check if the worklist entered already exists by calling method GET_WORKLIST_BY_WORKLIST_ID.
   If the worklist is found, store the worklist attributes for later update.
3. If the worklist does not exist yet:
   a. Call method *CREATE_WORKLIST* to create a new worklist. A worklist ID and a description have to be provided.
   b. Call method GET_WORKLIST_BY_WORKLIST_ID to read the attributes for later update.

4. Call method *ADD_ITEM* to add items to a worklist. Item key is the combination of worklist ID and object ID.
5. Call method CHANGE_WORKLIST with the updated attributes LAST_ITEM_CHANGE_BY and LAST_ITEM_CHANGE_AT (provide complete worklist structure).
6. Call method CHECK_BEFORE_SAVE to perform a validity check on all changes.
7. Call method SAVE_BUFFER to write the changes to the database.

The main part of the report could look as follows:

```
DATA:
  gv_batnr          TYPE /gbt/batnr,
  gv_matnr          TYPE /gbt/matnr,
  gv_plant          TYPE /gbt/plant,
  gv_logsys         TYPE /gbt/logsys,
  gv_now            TYPE timestamp,
  gt_object_id      TYPE /gbt/t_object_id,
  gt_worklist_id    TYPE /gbt/t_worklist_id,
  gt_worklist       TYPE /gbt/t_wrklst_hdr,
  gt_worklist_itm   TYPE /gbt/t_worklist_itm,
  gv_failed         TYPE /plmb/spi_failed_ind,
  gt_message        TYPE /plmb/t_spi_msg.

FIELD-SYMBOLS:
  <gs_worklist>     LIKE LINE OF gt_worklist,
  <gs_object_id>    LIKE LINE OF gt_object_id,
  <gs_worklist_itm> LIKE LINE OF gt_worklist_itm.


SELECT-OPTIONS:
  so_batnr FOR gv_batnr,
  so_matnr FOR gv_matnr,
  so_plant FOR gv_plant,
  so_lgsys FOR gv_logsys.

SELECTION-SCREEN SKIP 1.

PARAMETERS:
  p_wl_id  TYPE /gbt/w_wrklst_i-worklist_id,
  p_wl_dsc TYPE /gbt/w_wrklst_h-worklist_descr.


GET TIME STAMP FIELD gv_now.


*---------------------------------------------------------------------*
* Perform your selection to receive object IDs
* using the selection criteria so_batnr, so_matnr, ...
*---------------------------------------------------------------------*
PERFORM get_bat_guids CHANGING gt_object_id.

*---------------------------------------------------------------------*
* Create new worklist and read worklist attributes
*---------------------------------------------------------------------*
* Check if worklist already exists
APPEND p_wl_id TO gt_worklist_id.
/gbt/cl_worklist=>get_worklist_by_worklist_id(
  EXPORTING
    it_worklist_id  = gt_worklist_id
    iv_include_auth = abap_false
  IMPORTING
    et_worklist     = gt_worklist ).

IF lines( gt_worklist ) IS INITIAL.
* Worklist does not yet exist -> Create
  APPEND INITIAL LINE TO gt_worklist ASSIGNING <gs_worklist>.
```

```
  <gs_worklist>-worklist_id    = p_wl_id.
  <gs_worklist>-worklist_descr = p_wl_dsc.
  /gbt/cl_worklist=>create_worklist( gt_worklist ).

* Read new worklist for later update
  CLEAR gt_worklist.
  /gbt/cl_worklist=>get_worklist_by_worklist_id(
    EXPORTING
      it_worklist_id  = gt_worklist_id
      iv_include_auth = abap_false
    IMPORTING
      et_worklist     = gt_worklist ).
ENDIF.

*---------------------------------------------------------------------*
* Add objects as items to worklist
*---------------------------------------------------------------------*
* Build worklist item table
LOOP AT gt_object_id ASSIGNING <gs_object_id>.
  APPEND INITIAL LINE TO gt_worklist_itm ASSIGNING <gs_worklist_itm>.
  <gs_worklist_itm>-worklist_id = p_wl_id.
  <gs_worklist_itm>-obj_guid    = <gs_object_id>-obj_guid.
  <gs_worklist_itm>-obj_type    = <gs_object_id>-obj_type.
ENDLOOP.

* Write new items to buffer
/gbt/cl_worklist=>add_item( gt_worklist_itm ).

* Update items last changed by/at in worklist header for new items
LOOP AT gt_worklist ASSIGNING <gs_worklist>.
  <gs_worklist>-last_item_change_at = gv_now.
  <gs_worklist>-last_item_change_by = sy-uname.
ENDLOOP.

* Write changed worklist headers to buffer
/gbt/cl_worklist=>change_worklist( gt_worklist ).

* Check before save
/gbt/cl_worklist=>check_before_save(
  IMPORTING
    ev_failed  = gv_failed
    et_message = gt_message ).

* Save
/gbt/cl_worklist=>save_buffer( ).


*&---------------------------------------------------------------------*
* Read and display worklist & worklist items if wanted
*&---------------------------------------------------------------------*
     ...
```

This kind of report can e.g. run over night to refresh a worklist once a day or triggered by a user action.

7.4 Custom actions in GBT Worklist

It is possible to trigger actions on Objects referenced in a Worklist. For this the BAdI /GBT/EX_UI_NW_ACTION to Define and Execute Actions for Objects in the Network needs to be implemented.

You have to define the following BAdI methods:

- With method GET_ACTION_INFO, you add an new entry to table CT_ACTION_DEF.

With this entry, you provide the following information about the action, which is implemented in the BAdI:

- EVENT_ID: Technical ID of the action

  This ID is used to define the filter of the BAdI implementation (see description for method EXECUTE_ACTION below).

- TEXT: Description of the action, which is displayed in the dropdown box in the toolbar

- INDEX: Use this parameter to define the sequence (ascending) in which the actions are displayed in the dropdown box.

- With method EXECUTE_ACTION, you implement the function used as an action.

  Input parameters:

  - IT_OBJECT_ID: This table contains the selected objects, identified by object type and object GUID.
  - IV_WORKLIST_ID this method offers the reference to the Worklist at all. Therefore it is possible to get the Worklist description and the Users assigned to.

  Output/Changing parameters:

  - CT_MESSAGES: Use this table to return error or success messages.
    You have to fill the field SEVERITY and either fields MSGID and MSGNO for T100 messages, or simply field PLAINTEXT with a class text.

Example to get short text from the worklist:

```
IF iv_worklist_id IS NOT INITIAL. "Check if Action used from a Worklist
"   Read worklist details to get Short text
    APPEND iv_worklist_id TO lt_worklist_id.
    /gbt/cl_worklist=>get_worklist_by_worklist_id(
      EXPORTING
        it_worklist_id  = lt_worklist_id
        iv_include_auth = abap_false
      IMPORTING
        et_worklist     = lt_worklist ).
    READ TABLE lt_worklist INTO ls_worklist INDEX 1.
  ENDIF.
```

8 Enhancement of Network Graphic

8.1 Modification of node label and mouse overview

There are 3 ways to modify the graphical node content and mouse over information:

1. Use a custom DDIC structure: Within transaction SPRO go to 'SAP Global Batch Traceability'->'Business Add-Ins (BAdIs) for SAP GBT'->'User Interface'->'Modify UI Details of Object Types'. There you enter your own Z-structure (subset of structure /GBT/S_*_NODE_STRUCT) per object type. This is usually a copy of standard structure /GBT/S_*_UI_LABEL/OVERVIEW. For batch nodes you can even include class attributes by name if you have extended structure /GBT/S_BAT_NODE_STRUCT with those attributes. These fields are automatically populated in the order of the DDIC structure. You don't need to do anything further.
2. Extend standard structures /GBT/S_*_UI_LABEL/OVERVIEW with a custom include with a subset of structure /GBT/S_*_NODE_STRUCT. These fields are automatically populated in the order of the DDIC structure. You don't need to do anything further.
3. Use method DATA_MODIFY of BAdI /GBT/EX_UI_NW_GFX to modify/append fields. The order of the fields match the order of above mentioned structure /GBT/S_*_UI_LABEL/OVERVIEW with * being the technical object type (except plant / storage location and quantity / unit are concatenated into one field).

The SAP recommmended way is 1 or 2. Only for appending fields which are not part of /GBT/S_*_NODE_STRUCT you may want to use 3.

8.2 BAdI /GBT/EX_UI_NW_GFX

The BAdI /GBT/EX_UI_NW_GFX is used to modify all data shown in the graphical view of the global batch traceability network and to add additional callback functions in the context menu.

The following BAdI methods exist:

- DATA_MODIFY modifies the following data in the graphical view:
    - Network node instances including labels and colors
    - Network edge instances
    - Texts of buttons and context menus
    - Events of custom buttons and custom entries of network node context menus, and network edge context menus
- CUSTOM_GENERIC_EVENT_PROCESS, handles custom button callbacks.
- CUSTOM_NODE_EVENT_PROCESS, handles custom entries in network node context menus.
- CUSTOM_EDGE_EVENT_PROCESS, handles custom entries in network edge context menus.
- CLEANUP, clean up non-local modifications, for example, static class variables or function group variables, when navigating away from the graphical view. Other views, for example, the top-down or bottom-up table view, are not affected as a result.

Note: Constants for fields of parameters of method DATA_MODIFY can be found as attributes of interface /GBT/IF_EX_UI_NW_GFX. For example, /GBT/IF_EX_UI_NW_GFX=>GC_SETTING_NODEZOOM_MAX is used for altering the maximum node size via parameter CT_SETTINGS. The default value for this setting is 65 (percent).

9 Important GBT Classes and Examples

Disclaimer:

There is currently no class interface which is open to public use. Nevertheless programming enhancements may need to access GBT data. The following will give an overview about important internal used classes. But they may change without any warning if GBT development needs to adjust them.

Access to the Network: /GBT/CL_NW_READ

Access and Maintain GBT-Events, Ties, and Connectors: /GBT/CL_NW_DB

Access and maintain objects used as Network Node and Connectors: /GBT/CL_TRC_BO_<...>

Generic access to Objects used as Network Node and for Connectors: /GBT/CL_TRC_OBJ

Access to Master Data Objects: /GBT/CL_<...>_BO (Instantiation by /GBT/CL_*_FACTORY)

Info about connected Systems: /GBT/CL_SYS_INFO

9.1 Example for Maintain and Extract methods

Let's assume the Table /GBT/O_BAT_ITM was enhanced by ZZQUAN and ZZUOM. This should cumulate the quantity delivered from a specific Vendor. We need to access the node data and to update existing data of a node. The following coding example shows the usage of /GBT/CL_TRC_BO_<...> in the context of BAdI /GBT/EX_IL_EVENT_ENHANCE by implementing the method `EVENT_ENHANCE`.

The example covers the methods:

```
/gbt/cl_trc_bo_<...>=>get_single_itm_by_key – getting a single data set
/gbt/cl_trc_bo_<...>=>extract_itm – get a table of data sets in maintain structure
/gbt/cl_trc_bo_<...>=>maintain – set a table of data sets for update/insert
/gbt/cl_trc_obj=>save – do a save to all changed node and Connector Objects
METHOD /gbt/if_ex_il_event_enhance~event_enhance.
  FIELD-SYMBOLS: <ls_event_maintain> LIKE LINE OF ct_event_maintain.
  DATA: lt_puri_key TYPE /gbt/t_puri_key,
        lt_bat_itm_key TYPE /gbt/t_bat_itm_key.
  FIELD-SYMBOLS: <ls_puri_key> TYPE /gbt/s_puri_key,
                 <ls_bat_itm_key> TYPE /gbt/s_bat_itm_key.
  DATA: ls_puro_hdr TYPE /gbt/s_puro_hdr_data,
        lt_bat_maint_itm TYPE /gbt/t_bat_maint_itm.

  FIELD-SYMBOLS: <ls_bat_maint_itm> TYPE /gbt/s_bat_maint_itm.

  " 1. Get Document for PUrchase order Item (PUI) (filter by Vendor)
and BATch(BAT)
  LOOP AT ct_event_maintain ASSIGNING <ls_event_maintain>.
    IF <ls_event_maintain>-obj_type_from = 'PUI' AND
       <ls_event_maintain>-obj_type_to = 'BAT'.
      CLEAR: lt_puri_key, lt_bat_itm_key.

      " Get Vendor of Purchase Order
      ASSIGN <ls_event_maintain>-r_obj_key_from->* TO <ls_puri_key>.
      /gbt/cl_trc_bo_puri=>get_single_itm_by_key(
        EXPORTING
          is_key       = <ls_puri_key> " Key for Purchasing Document Item
        IMPORTING
          es_hdr_data = ls_puro_hdr    " Data of Purchasing Document Head
er
```

```
      ).

      IF ls_puro_hdr-vendor <> 'VENDOR_A'. "Filter on A Vendor
        CONTINUE. "not needed
      ENDIF.

      " Get Quantity of Batch Item
      ASSIGN <ls_event_maintain>-r_obj_key_to->* TO <ls_bat_itm_key>.
      APPEND <ls_bat_itm_key> TO lt_bat_itm_key.
      /gbt/cl_trc_bo_bat=>extract_itm(
        EXPORTING
          it_item_key      =   lt_bat_itm_key  "
        IMPORTING
          et_item          =   lt_bat_maint_itm  "
      ).
      READ TABLE lt_bat_maint_itm ASSIGNING <ls_bat_maint_itm> INDEX 1.
      IF sy-subrc = 0.
        " 2. take the Quantity & UoM from MDoc Item (alternative use Quan
tity of Event)
        IF <ls_bat_maint_itm>-zzuom IS NOT INITIAL.
          IF <ls_bat_maint_itm>-zzuom <> <ls_event_maintain>-unit.
            CONTINUE. "something wrong-what to do???
          ENDIF.
        ELSE.
          <ls_bat_maint_itm>-zzuom = <ls_event_maintain>-
unit. "initialize UoM
        ENDIF.

        " calculate new Quantity
        " Attention! Unsolved so far in this
example: how to detect if the Quantity is already added (for Reload of ex
iting Events this is wrong)
        IF <ls_event_maintain>-db_cr_ind = 'S'.
          "add quantity
          <ls_bat_maint_itm>-zzquan = <ls_bat_maint_itm>-
zzquan + <ls_event_maintain>-quantity.
        ELSE. " Cancelation
          "substract quantity
          <ls_bat_maint_itm>-zzquan = <ls_bat_maint_itm>-
zzquan - <ls_event_maintain>-quantity.
        ENDIF.
      ELSE.
        CONTINUE. "something wrong-what to do???
      ENDIF.

      " 3. Store / update / maintain  Batch at Location (Plant/Stge_Loc)
(/GBT/O_Bat_Itm (extended Fields)) with the Quantity.
      /gbt/cl_trc_bo_bat=>maintain(
        EXPORTING
          it_item     =   lt_bat_maint_itm  " Batch Maintenance for inter
nal tracking member
      ).
    ENDIF.
  ENDLOOP.

  " Save Bat Item by saving all changes to Network Objects.
  /gbt/cl_trc_obj=>save( ).
ENDMETHOD.
```

9.2 Example for setting appended fields

The example covers extension of the method /GBT/IF_TRC_OBJ~GET_UI_DATA_BY_GUID by an Implicit Enhancements.

1. Append 2 fields *zzsold_to_name1 TYPE bu_mcname1 and zzsold_to_name2 TYPE bu_mcname2 to structure* /GBT/S_SALI_NODE_STRUCT.
2. Do an implicit enhancement of /GBT/CL_TRC_BO_SALI->/GBT/IF_TRC_OBJ~GET_UI_DATA_BY_GUID and add at the end of the method the following logic to set *zzsold_to_name1 and zzsold_to_name2:*

```
DATA:
      lo_prt_bo             TYPE REF TO /gbt/cl_prt_bo.
FIELD-SYMBOLS:
<ls_sali_node_struct> TYPE /GBT/S_SALI_NODE_STRUCT.

*appended:
zzsold_to_name1 TYPE bu_mcname1 zzsold_to_name2 TYPE bu_mcname2

*Get the GBT Partner class for Customer
lo_prt_bo ?= /gbt/cl_prt_factory=>get_instance( iv_gos_otype_name = /gbt/
cl_prt_bo=>cv_gos_otype_customer ).

LOOP at et_data assigning <ls_sali_node_struct> .

lo_prt_bo->/gbt/if_prt_bal~get_name(
  EXPORTING
    iv_prtnr  =   <ls_sali_node_struct>-sold_to " ERP Customer Number
    iv_logsys =   <ls_sali_node_struct>-logsys  " Logical System
  IMPORTING
    ev_name1  = <ls_sali_node_struct>-zzsold_to_name1
    ev_name2  = <ls_sali_node_struct>-zzsold_to_name2
).
" missing name => show key instead
IF <ls_sali_node_struct>-zzsold_to_name1 IS INITIAL AND
   <ls_sali_node_struct>-zzsold_to_name2 IS INITIAL.
  /gbt/cl_fmt_util=>encapsulate_string(
    EXPORTING
      iv_info              = <ls_sali_node_struct>-sold_to
    IMPORTING
      ev_info_encapsulated = <ls_sali_node_struct>-name1
  ).
ENDIF.
ENDLOOP.
```

10 Use of Implicit Enhancements

Implicit enhancements can be placed at start and end of methods and function modules.

The implicit enhancement options can be displayed in the ABAP Editor by following the path: *Edit* -> *Enhancement Operations* -> *Show Implicit Enhancement Options*, and then enhanced using source code plug-ins. You directly put your cursor on the enhancement option and create your implementations by right click. See also http://wiki.sdn.sap.com/wiki/display/ABAP/How+To+Do+Implicit+Enhancement .

Attention

There is a risk to use implicit enhancements. Please be aware that coding might change and that this also may affect the method interface or use of this interface. Also the method might be called at different places which not be expected before.

11 SOA Service Enhancement

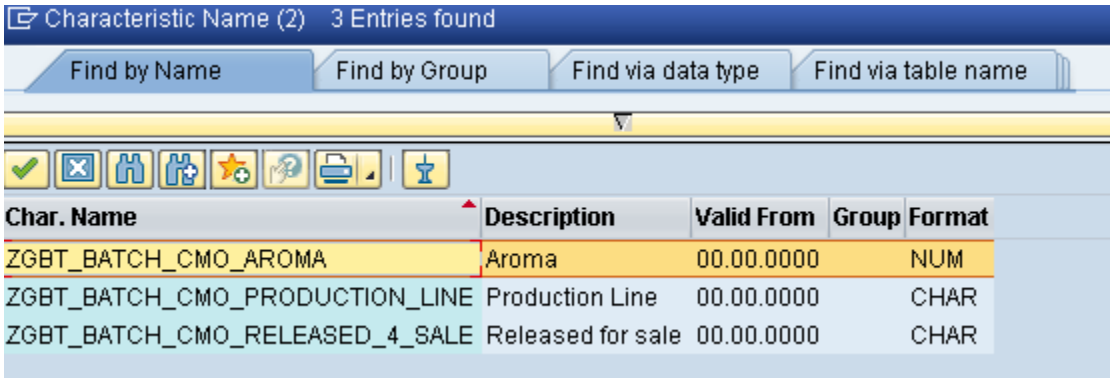
11.1 SOA Inbound Services

The SOA service interfaces supports also customer enhancements. It is possible to apply new data fields or even complete new data nodes. The procedure of Enterprise Service enhancement is described in the official available SAP Enterprise Service Enhancement Guide 2.0 (http://scn.sap.com/docs/DOC-18402).

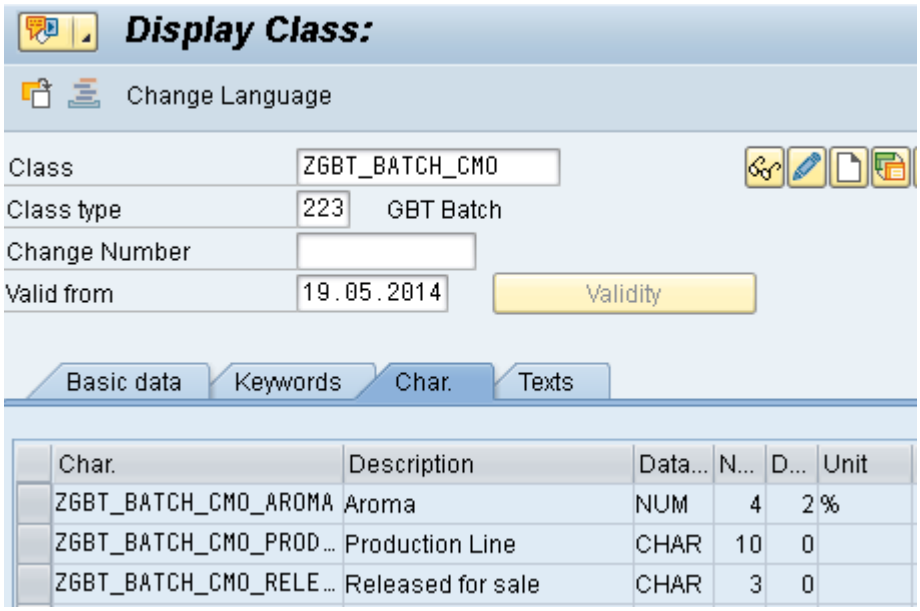
There exist the possibility to enhance the standard SOA interfaces for master data and the goods movement event data. Also for the standard identification of a sender party can be enhanced or overruled. Therefore

As a good example and known practice, the below example shows how the **TrackedGoodsMovementNotification_In** (Batch-related event data import) service interface will be enhanced by adding new data fields to classify batches. The steps below describe the necessary development actions in detail. You have to do two different activities in two different systems. These are the SAP Enterprise Service Repository (ESR) and the SAP BGT ABAP backend. If you do not know the URL to the SAP ESR you can use the transaction SPROXY to launch the SAP Exchange Infrastructure in the SAP GBT backend.

1. Firstly, it is necessary to create a customer own software component version and namespace in the SAP ESR, as described in detail in the Enterprise Service Enhancement Guide 2.0. In the ESR you find also the technical name of the data type you want to enhance. In our example it is the data type TrackedGoodsMovmntBatchObj (element node for batch).
2. Clarify which additional fields are necessary for the batch. In our example we use new attributes like ProductionLine, Aroma and ReleasedForSales. These attributes need to be created as characteristics with ABAP transaction CT04 before.

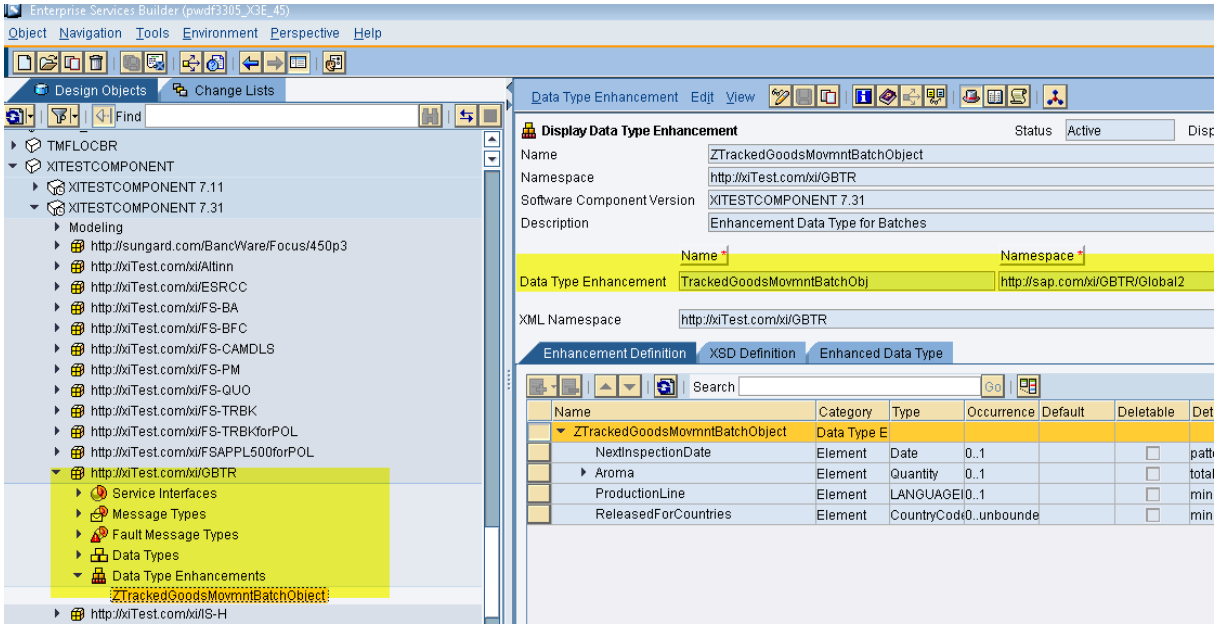


Screen-shot: Additional characteristics



Screen-shot: Assingment of the characterstics to a class

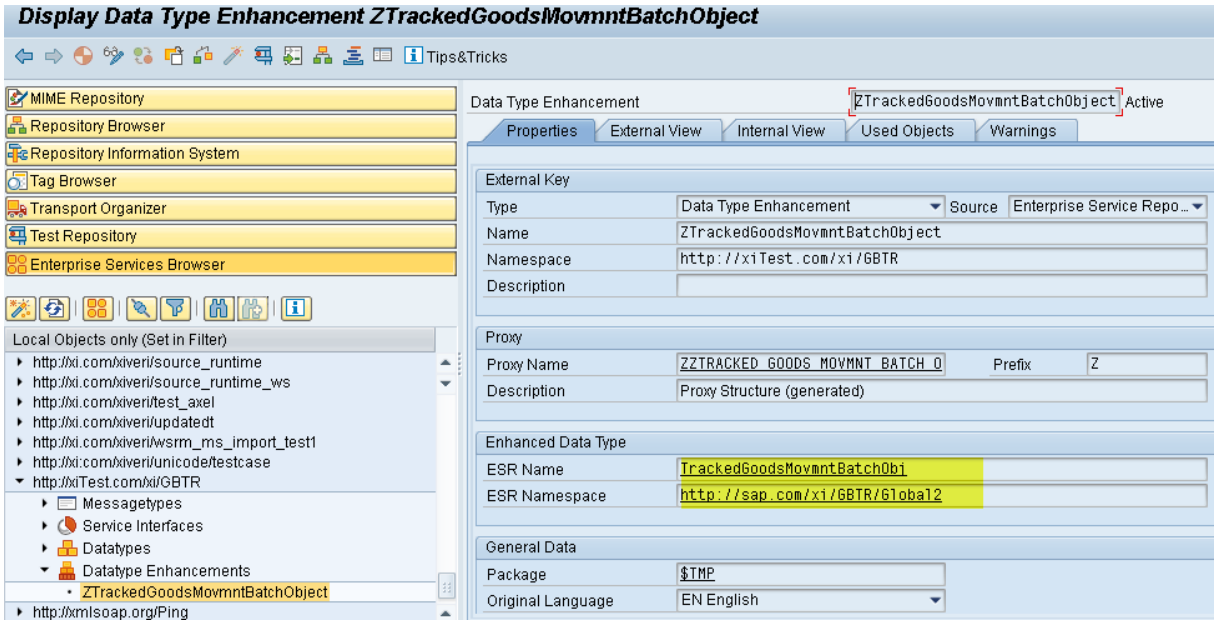
3. Assign the newly created characteristics to an existing classification class or create a new one. In our example we use a new class with name ZGBT_BATCH_CMO and class type '223'.
4. Enhance the data type TrackedGoodsMovmntBatchObj in the SAP ESR with the new characteristics you created in the step before. This can be done by creating a data type enhancement, which refers to the data type TrackedGoodsMovmntBatchObj. Assign this data type enhancement to your own created software component version and namespace, created in step 1.
5. Check and activate the data type enhancement.



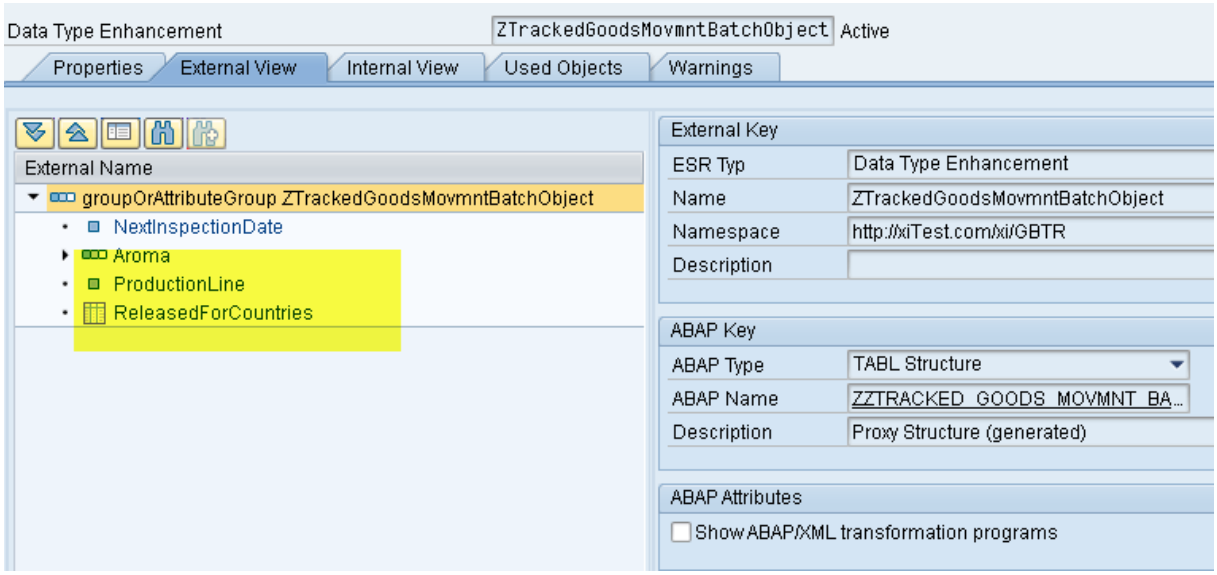
Screen-shot: Data type Enhancement in the SAP ESR

6. Go to the SAP GBT ABAP backend system. Start transaction SPROXY.
7. Navigate to your software component version and namespace. Double-click on the data type enhancement and regenerate the data type enhancement. Save and active it. During the activation

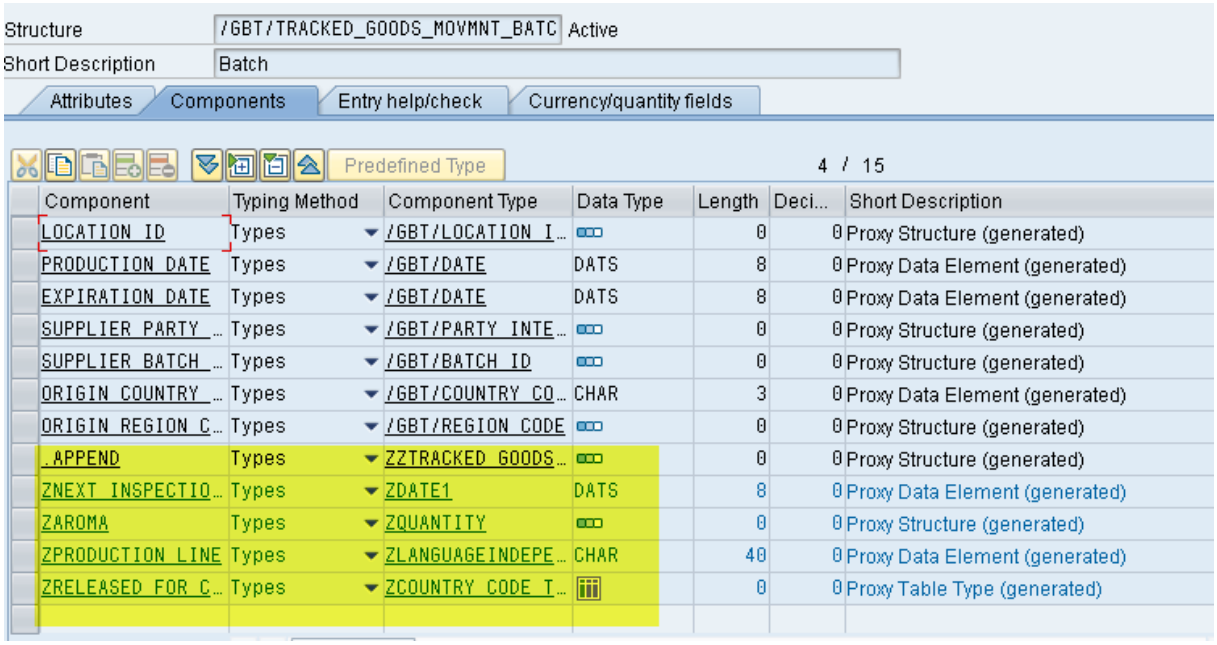
phase the existing ABAP DDIC proxy of the data type TrackedGoodsMovmntObj is enhanced by a append structure which consists the new attributes.



Screen-shot:Tthe data type enhancement generated in the SAP GBT backend (transaction SPROXY)



Screen-shot: Structure of the data type enhancement



Screen-shot: Append structure of the data type enhancement to the origin data type

8. In the last step a BADI implementation for interface /GBT/IF_EX_IL_B2B_TRC_GDS_MVM needs to be created, which maps the incoming batch data to the API from SAP GBT. Therefore, the corresponding interface method /GBT/IF_EX_IL_B2B_TRC_GDS_MVM~MAP_BATCH has to be implemented. Below you find a code example.

```
METHOD /gbt/if_ex_il_b2b_trc_gds_mvm~map_batch.
  FIELD-SYMBOLS: <lf_batch> TYPE /gbt/tracked_goods_movmnt_batc,
                 <lf_bat_maint> TYPE /gbt/s_bat_maint,
                 <lf_rel_4_country> TYPE zcountry_code.

DATA:
ls_clf_val_num      LIKE LINE OF <lf_bat_maint>-clf-t_val_num,
ls_clf_val_char     LIKE LINE OF <lf_bat_maint>-clf-t_val_char.

*->Customer Enhancements for batches
LOOP AT it_batch ASSIGNING <lf_batch>.
READ TABLE ct_bat_maint ASSIGNING <lf_bat_maint> INDEX sy-tabix.
    IF sy-subrc = 0.
      "NextInspectionDate
      <lf_bat_maint>-nextinspec_date = <lf_batch>-
znext_inspection_date.
      "Classifiction Characteristics
      <lf_bat_maint>-clf-s_alloc-classnum    = 'ZGBT_BATCH_CMO'.
      <lf_bat_maint>-clf-s_alloc-classtype   = '223'.
      <lf_bat_maint>-clf-s_alloc-objecttable = '/GBT/BATCH'.
      "--Aroma
      IF <lf_batch>-zaroma-content IS NOT INITIAL.
        ls_clf_val_num-charact               = 'ZGBT_BATCH_CMO_AROMA'.
        ls_clf_val_num-unit_from        = <lf_batch>-zaroma-unit_code.
        ls_clf_val_num-value_from        =  <lf_batch>-zaroma-content.
        APPEND ls_clf_val_num TO <lf_bat_maint>-clf-t_val_num.
        CLEAR ls_clf_val_num.
      ENDIF.
      "--Production line
      IF <lf_batch>-zproduction_line IS NOT INITIAL.
        ls_clf_val_char-charact
```

```
                        = 'ZGBT_BATCH_CMO_PRODUCTION_LINE'.
          ls_clf_val_char-value_char       = <lf_batch>-zproduction_line.
          APPEND ls_clf_val_char TO <lf_bat_maint>-clf-t_val_char.
          CLEAR ls_clf_val_char.
        ENDIF.
        "--Released for sale
        LOOP AT <lf_batch>-zreleased_for_countries
                                          ASSIGNING <lf_rel_4_country>.
          ls_clf_val_char-charact
                        = 'ZGBT_BATCH_CMO_RELEASED_4_SALE'.
          ls_clf_val_char-value_char  = <lf_rel_4_country>.
          APPEND ls_clf_val_char TO <lf_bat_maint>-clf-t_val_char.
          CLEAR ls_clf_val_char.
        ENDLOOP.

      ENDIF.
    ENDLOOP.
ENDMETHOD.
```

9. Test the service interface TrackedGoodsMovementNotification_In within the test environment (F8) of the transaction SPROXY. Below a xml snippet is depicted, which supports the enhanced attributes in the batch element node.

```
…
        <Batch>
          <ID>2002_1</ID>
          <MaterialID>GBT_FINISH1</MaterialID>
          <LocationID>0001</LocationID>
          <SubLocationID>0001</SubLocationID>
          <ExpirationDate>2020-02-22</ExpirationDate>
          <OriginCountryCode>CA</OriginCountryCode>
          <OriginRegionCode>ON</OriginRegionCode>
          <gbtr:NextInspectionDate>2015-02-19</gbtr:NextInspectionDate>
          <gbtr:Aroma unitCode="%">19.15</gbtr:Aroma>
          <gbtr:ProductionLine>LINE_B</gbtr:ProductionLine>
          <gbtr:ReleasedForCountries>DE</gbtr:ReleasedForCountries>
          <gbtr:ReleasedForCountries>FR</gbtr:ReleasedForCountries>
        </Batch>
…
```

11.2 SOA Outbound Service (DRAFT)

With SAP GBT 2.0 FP02 the SAP GBT offers also an SOA outbound web service interface. This outbound web service interface will be called via an example report, which select the top-down event data by deliveries. In brief, the outbound web service interface uses the same message type (data structure) as the inbound web service interface. This means that, when the inbound web service interface will be structurally enhanced then in the same way the outbound web service interface is also be enhanced.

The outbound web service also provide BAdI's to enhance the business data provisioning for the enhanced outbound web service interface. So it is possible to enhance or change the event key and the business object details mapping.

The report /GBT/R_SEND_TRCD_FOR_DEL_GOODS is an example implementation, which shows how the outbound web service interface can be used. It contains the basic steps like authorization check,

selecting GBT event data, mapping of internal to external data, applying ID tagging information and finally sending of the transformed SAP GBT data to an external system.

11.2.1 Enhancement of the identifier tagging

The outbound service provide the functionality to change the tagging of used identifiers in the sending message. To avoid identifier (ID) clashes of business objects, which potentially came from different ERP systems, it is necessary provide an ID tagging. The ID tagging provide the information about the issuer of the ID. The ID tagging uses the standard SAP GDT attributes for Identifiers like Scheme Agency Id and Scheme ID.

The SAP GBT 2.0 FP02 delivers three different standard implementations to provide an identifier tagging. These are ID tagging on logical system, SLD business system ID and Global Location Number for plants.

ABAP class	Description
/GBT/CL_TRCD_IDS_LOGSYS	Identifier Key Mapping for Logical System
/GBT/CL_TRCD_IDS_SLD_BS_ID	Identifier Key Mapping for SLD Business Key
/GBT/CL_TRCD_IDS_PLANT_GLN	Identifier Key Mapping for Plant Global Location Number

In the SAP GBT standard ID tagging we use the ID tagging only on the first key level. This means for example for batches, which consists of a concatenated key like batch, material and plant number we provide the ID tagging only for the batch number in the concatenated key. This is sufficient to uniquely identify objects in the overall message content. The same is true for all other objects like Handling Unit and Document References (Purchase Order, Sales Order, Delivery,...).Below a xml snippet is shown, which uses for example the SLD Business Key ID mapping.

```
…
<GoodsMovementEvent>
  <From>
    <BatchDocumentReference>
        <ID schemeAgencyID="CMO_001">2002_1</ID>
        <MaterialID>GBT_RAW1</MaterialID>
        <LocationID>0001</LocationID>
        <SubLocationID>0001</SubLocationID>
    </BatchDocumentReference>
  </From>
<To>
  <DocumentReference>
     <ID schemeAgencyID="CMO_001">1001081</ID>
    <TypeCode>97</TypeCode>
    <CreationDate>2014-01-01</CreationDate>
  </DocumentReference>
</To>
…
```

If the customer or partner needs a special ID tagging, an interface is provided, which allows an own implementation.

In a public constructor of the own ID tagging provider ABAP class, it is necessary to provide an own scheme ID and a description for that ID, as depicted in the code snippet below:

```
method CONSTRUCTOR.
  /GBT/IF_TRCD_IDS~GV_SCHEMA       = 'ZSCHEMA_ID'.
  /GBT/IF_TRCD_IDS~gv_description =  text-001.
endmethod.
```

The interface is called:

/GBT/IF_TRCD_IDS and has the following methods and attributes:

- GET_KEYS: Map internal to external keys

Parameter	TYPE	Data Type	Description
IS_KEY_PROVIDER	Importing	/GBT/S_TRCD_KEY_PROVIDER	Internal Key Provider like logical system, plant,...
IS_KEY_CONTEXT	Importing	/GBT/S_TRCD_KEY_CONTEXT	Identifier context
IV_VALUE_INT	Importing	CSEQUENCE	Internal Key Value
ES_VALUE_EXT	Exporting	/GBT/S_TRCD_IDS	External Key Values (SchemeAgencyID, SchemeID and Content)

- CLEAR_BUFFER: Clear internal key mapping buffer

To provide an own implementation for ID tagging, it is necessary to implement this interface methods in an own ABAP class. A factory class is available to create an instance as a singleton.

/GBT/CL_TRCD_IDS_FACTORY Factory class for Tracked Goods Movement Identifier Tagging

- GET_INSTANCE: Get instance of identifier tagging provider

Parameter	TYPE	Data Type	Description
IV_SCHEME_ID	Importing	/GBT/SCHEME_ID	Scheme ID like SAP_LOGSYS, SAP_SLD_BSID, SAP_PLANT_GLN, etc.
ER_ID_SCHEME	Exporting	/GBT/IF_TRCD_IDS	External Key Providers for Identifiers

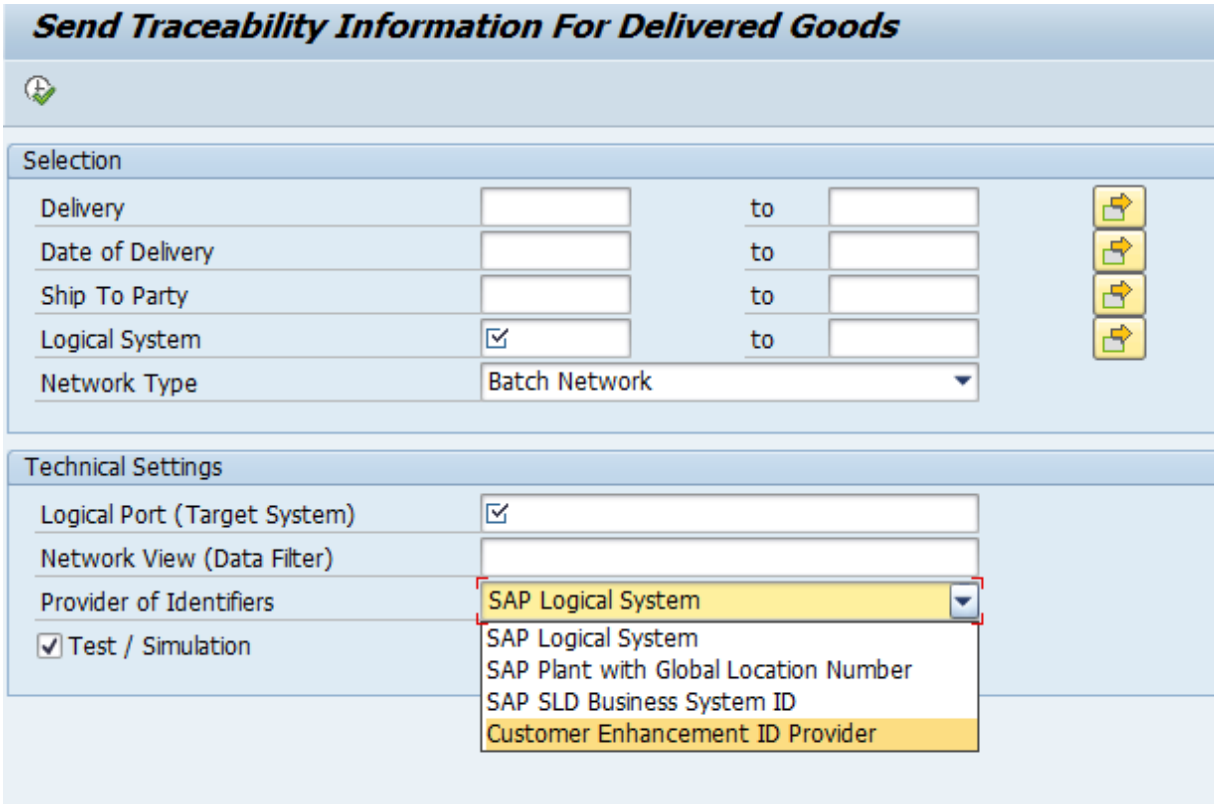
- GET_SCHEMES: Get meta information of all schema providers

Parameter	TYPE	Data Type	Description
ET_SCHEME	Exporting	/GBT/CL_TRCD_IDS_FACTORY=>TT_SCHEME_ID	List of schemes of ID provider

Attributes and constants:

Attribute	Level	Data Type	Description
GV_SCHEMA	Instance	/GBT/SCHEME_ID	Schema ID
GV_DESCRIPTION	Instance	/GBT/DESCR	Schema description
CV_SAP_LOGSYS	Constant	/GBT/SCHEME_ID	SAP_LOGSYS
CV_SAP_SLD_BS_ID	Constant	/GBT/SCHEME_ID	SAP_SLD_BS_ID
CV_SAP_PLANT_GLN	Constant	/GBT/SCHEME_ID	SAP_PLANT_GLN

The ID tagging is used in the example report mentioned above. The selection screen of this report is depicted below:



11.2.2 Enhancement of the TrackedGoodsMovementNotification_Out interface

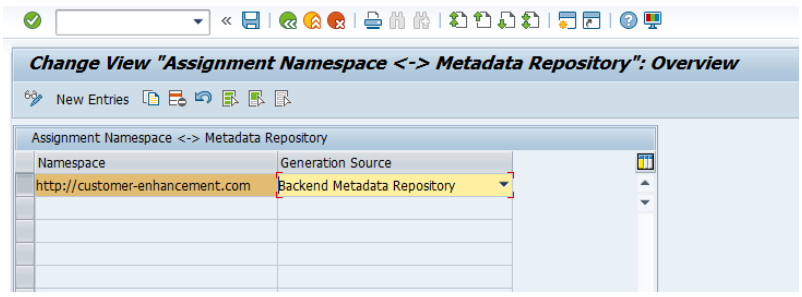
In chapter 11.1 above the process is described how to enhance the SOA inbound service interface when the SAP XI, as a so called middleware, is used in the customer system landscape to control and orchestrate centrally the overall message flow.

HINT: The inbound and the outbound SOA web service interface use the same message type. This means, when you enhance the inbound service interface, also the outbound interface is enhanced accordingly and vice versa.

In this chapter another possibility to enhance SOA service interfaces is described, when no SAP XI is used at customer side.

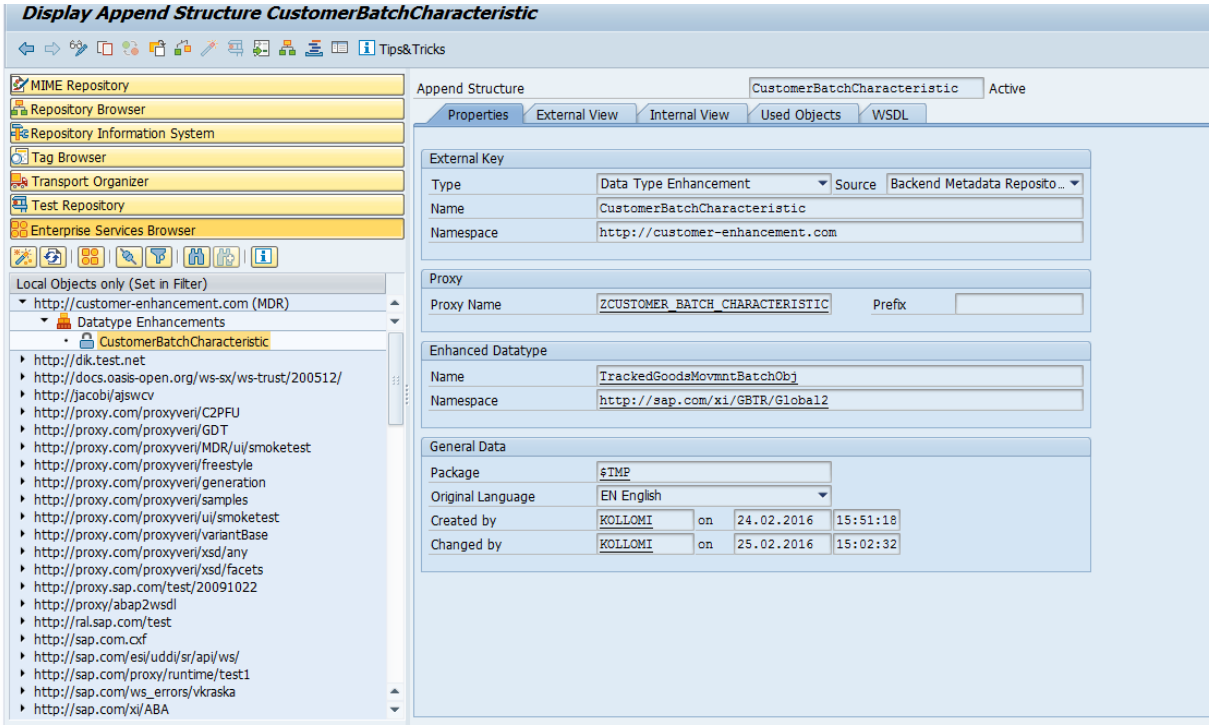
The step-by-step description below, supports the use case that existing classification data needs to be exposed by the outbound web service interface on the batch detail level.

1. Register customer namespace with transaction **SPXNGENAPPL**, for instance namespace = http://customer-enhancement.com and Generation Source = Backend Metadata Repository.

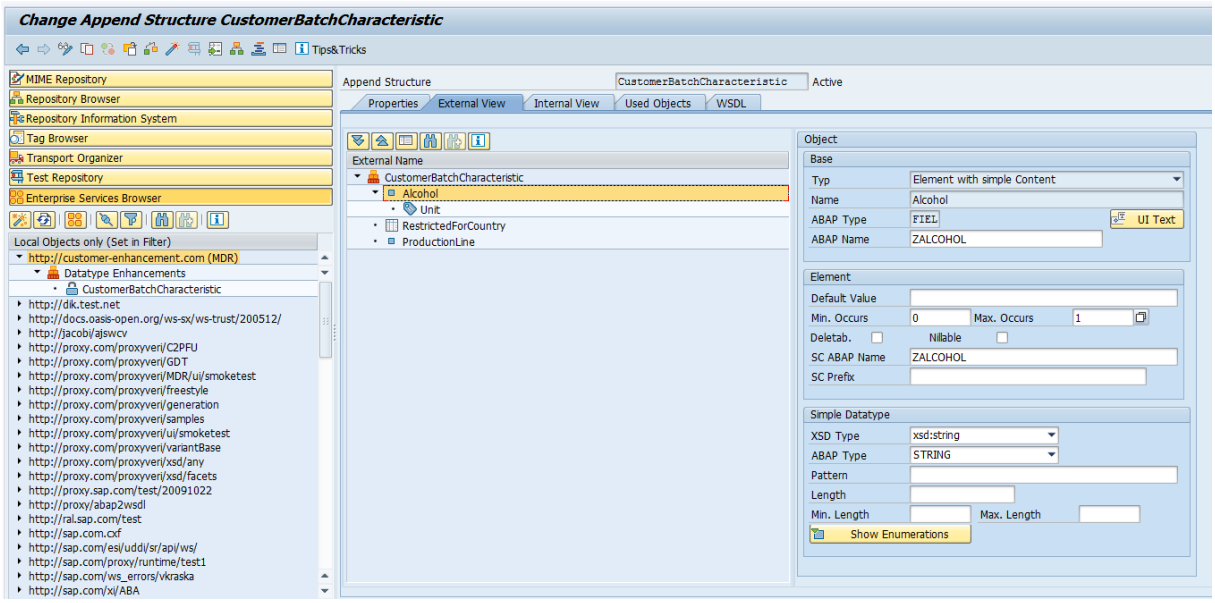


2. Start transaction **SPROXY**
3. Press on the button ‘Create MDR Proxy’ 
4. Choose option: Data type Enhancement and press continue.
5. Choose option: Backend and press continue.
6. Enter an own qualified name and namespace of the data type enhancement, for instance Name = ZBatchDetailsEnhancement and namespace = http://customer-enhancement.com.
7. Enter the qualified name and namespace of the data type, which needs to be enhanced, for instance name = TrackedGoodsMovmntBatchObj and namespace = http://sap.com/xi/GBTR/Global2. Press continue.
8. Choose a valid development package or store it local in the system. Choose a prefix, if you want and press continue.
9. Press button ‘Complete’ to finalize the data type enhancement creation.
10. Save and activate the data type enhancement.

Below, the activated data type enhancement is shown within the transaction SPROXY.

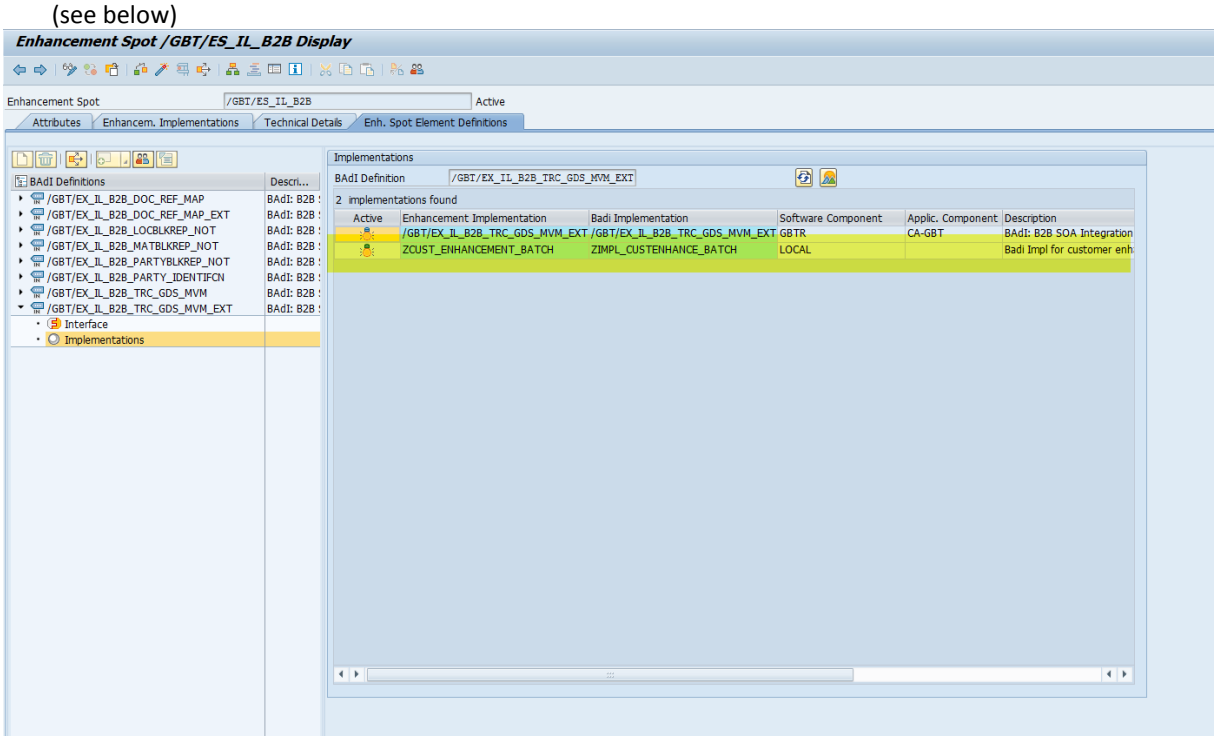


11. In the change-mode it is now possible to add arbitrary elements and attributes. Therefore it is necessary to switch to tab ‘External View’ or ‘Internal View’ and set the cursor to the top level node (here: for instance CustomerBatchCharacteristic) and open the context menu. In our example we add three new elements like Alcohol, Restricted for Country and Production Line (see below).

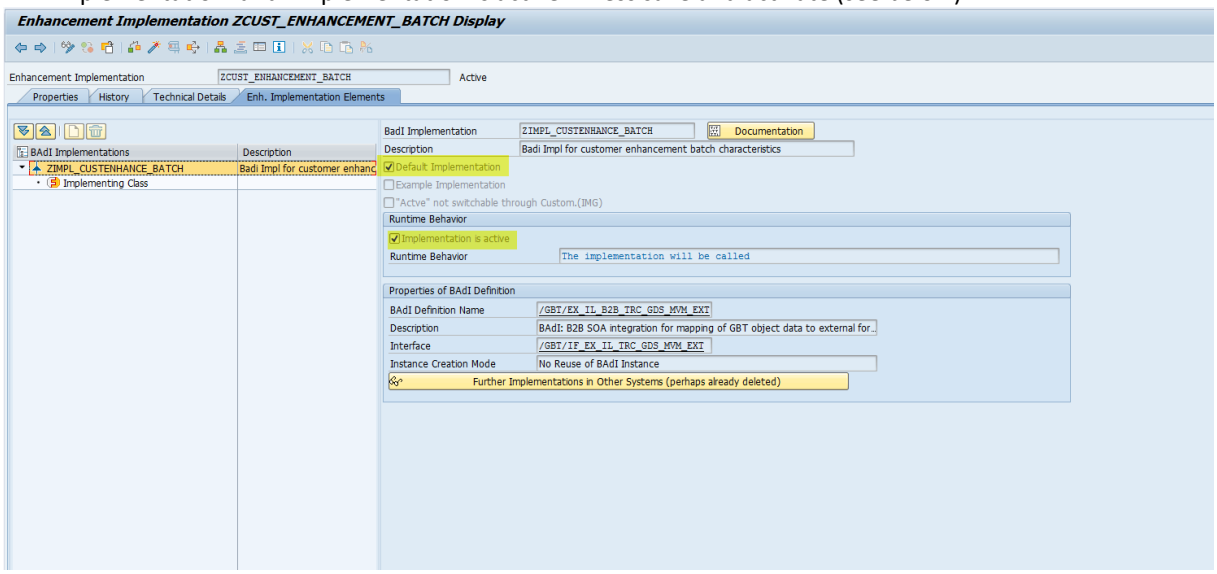


12. The next step is to provide a BADI implementation for the data type enhancement in the outbound service. The outbound service offers a BADI interface to provide access to the different node structures. In this example the batch detail node was enhanced, therefore the method MAP_BATCH needs to be implemented to map the GBT internal data to the external outbound web service interface data.
    Use transaction SE18 and choose the enhancement spot /GBT/ES_IL_B2B.

13. Right mouse-click on the implementation of BAdI-Definition /GBT/EX_IL_B2B_TRC_GDS_MVM_EXT and choose ‘Create BAdI-Implementation’.
14. Press the create button on the popup appearing.
15. Provide an arbitrary name (in a customer system something with a starting Z) of the Enhancement Implementation and a descriptive text. Press button ‘Enter’.
16. Choose a proper development package (here: Local Object). Press the button ‘Enter’.
17. Choose the currently created Enhancement Implementation and create an own BAdI Implementation.
18. Choose an arbitrary name (for instance: Customer Z name), a descriptive text and an Implementation Class (see below)



19. Double-click on the newly created BAdI implementation and mark the checkboxes for ‘Default Implementation’ and ‘Implementation is active’. Press save and activate (see below).



20. Double-click on the Implementation Class. Double-click on the method /GBT/IF_IL_TRC_GDS_MVM_EXT~MAP_BATCH. Provide a ABAP code implementation to read the classification data through the GBT API: `/gbt/cl_clf_bo=>get_clf_classification.` Below an example coding is depicted to provide the necessary information from GBT and the corresponding mapping from the internal to the external data representation. It is possible to copy and paste the complete coding. It is only necessary to adapt the class name and classification attributes according to your attributes.

```
METHOD /gbt/if_ex_il_trc_gds_mvm_ext~map_batch.
  DATA:
           lo_struc            TYPE REF TO cl_abap_structdescr,
           ls_bat_details      TYPE /gbt/tracked_goods_movmnt_batc,
           lv_class            TYPE klasse_d,
           lv_class_type       TYPE klassenart,
           ls_class            TYPE /gbt/s_class,
           lt_attr             TYPE /gbt/cl_attr=>tt_attr,
           lv_obj_guid         TYPE /gbt/obj_guid,
           ls_msg_list         TYPE /gbt/s_msg_list,
           lv_indx             TYPE sy-tabix.


  FIELD-SYMBOLS:
         <ls_obj_data>       TYPE /gbt/cl_nw_read=>ts_obj_type_data,
         <lt_data_tab>       TYPE ANY TABLE,
         <ls_obj_type_data>  TYPE any,
         <lv_comp>           TYPE any,
         <ls_comp_def>       TYPE abap_compdescr,
         <ls_attr>           LIKE LINE OF lt_attr,
         <ls_bat_detail>     LIKE LINE OF ct_bat_detail.

  READ TABLE it_event_obj_data WITH KEY obj_type = iv_gbt_obj_type  ASSIGNING <ls_obj_data>.
  IF sy-subrc = 0.

    ASSIGN <ls_obj_data>-data_tab->* TO <lt_data_tab>.

    LOOP AT <lt_data_tab> ASSIGNING <ls_obj_type_data>.
      CLEAR: lv_class, lv_class_type, lv_obj_guid.
      lv_indx = lv_indx + 1.

      AT FIRST.
        lo_struc ?= cl_abap_typedescr=>describe_by_data( <ls_obj_type_data> ).
      ENDAT.


      LOOP AT lo_struc->components ASSIGNING <ls_comp_def> WHERE name = 'CLASS_TYPE'  OR
                                                                 name = 'CLASS'       OR
                                                                 name = 'HDR_GUID'.
```

```
        ASSIGN COMPONENT <ls_comp_def>-name
                    OF STRUCTURE <ls_obj_type_data> TO <lv_comp>.

        CASE <ls_comp_def>-name.

          WHEN 'CLASS'.
            lv_class = <lv_comp>.

          WHEN 'CLASS_TYPE'.
            lv_class_type = <lv_comp>.

          WHEN 'HDR_GUID'.
            lv_obj_guid = <lv_comp>.

          WHEN OTHERS.

        ENDCASE.
      ENDLOOP.

      IF lv_class IS NOT INITIAL AND lv_class_type IS NOT INITIAL.


        /gbt/cl_clf_bo=>get_clf_classification( EXPORTING iv_object_tab =  '/GBT/BATCH'
                                                          iv_object_key =  lv_obj_guid
                                                          iv_class_type =  '223'
                                                IMPORTING es_class      =  ls_class
                                                          et_attr       =  lt_attr
                                                          es_msg_list   =  ls_msg_list ).
        IF ls_class-class = 'ZSOA_ENHANCEMENT01'.
          READ TABLE ct_bat_detail INDEX lv_indx ASSIGNING <ls_bat_detail>.
          IF sy-subrc <> 0.
            CONTINUE.
          ENDIF.

          LOOP AT lt_attr ASSIGNING <ls_attr>.
            CASE <ls_attr>-key.

              WHEN 'GBT_ALCOHOL'.
                <ls_bat_detail>-zalcohol-content = <ls_attr>-valn_id-valfr.
                <ls_bat_detail>-zalcohol-unit    = <ls_attr>-valn_id-vunit.

              WHEN 'GBT_PRODUCTION_LINE'.
                <ls_bat_detail>-zproduction_line  = <ls_attr>-valc_id.

              WHEN 'GBT_RESTRICTED_FOR_COUNTRY'.
                IF  <ls_attr>-valc_id IS NOT INITIAL.
                  APPEND <ls_attr>-valc_id TO <ls_bat_detail>-zrestricted_for_country.
                ENDIF.

              WHEN OTHERS.
            ENDCASE.
          ENDLOOP.
        ENDIF.

      ENDIF.
    ENDLOOP.

  ENDIF.

ENDMETHOD.
```