ISU BOL Tree: Configuration and Enhancement

Applies to:
SAP CRM 7.0. For more information, visit the Customer Relationship Management homepage.

Summary
This document is aimed at understanding the configuration and implementation for the ISU BOL tree. The document will also look at how to implement custom requirements for BOL tree profiles.

Author: Mamta Saboo
Company: Deloitte Consulting India Pvt. Ltd.
Created on: 16 June, 2011

Author Bio
Mamta Saboo is a Senior Consultant at Deloitte Consulting. She has worked on many CRM implementation projects. She is SAP certified in PI module.
# Table of Contents

ISU Tree Profiles .................................................................................................................. 3
Configuration of Tree Profiles ............................................................................................... 3
Tree Profile Attributes ............................................................................................................. 3
Definition of Columns ............................................................................................................. 4
Tree Profile Hierarchy .............................................................................................................. 5
Comparison of Custom Profile with Standard Tree Profile ..................................................... 6
Node Details and its Integration with Custom Development ..................................................... 7
Custom Development Associated with BOL Tree .................................................................... 8
Configuration of Presentation IDs .......................................................................................... 9
One Click Actions ..................................................................................................................... 11
Assign tree Profiles to Overview ............................................................................................ 12
Implementation of Generic Tree Component IUBOTree .......................................................... 13
  Enhancement in Parent Component ...................................................................................... 13
  Tree Properties ................................................................................................................... 14
  Event handling in tree ......................................................................................................... 14
  OCA and links ..................................................................................................................... 16
  Toolbar buttons .................................................................................................................. 16
Related Content ...................................................................................................................... 17
Disclaimer and Liability Notice ............................................................................................... 18
ISU Tree Profiles
The tree profile is a widely used feature in CRM IS-Utilities implementations. It gives the flexibility of re-using the component in various other components.

Configuration of Tree Profiles
Path in SPRO:
Customer Relationship Management -> Industry-Specific Solutions -> Telecommunications -> Master Data -> Object Management -> Settings for Object Workbench -> Define Object Hierarchy and Object Attributes
Or Start View Cluster CRM_IU_IC_BOLTREE in transaction SM34.

View Cluster Maintenance: Initial Screen

View cluster CRM_IU_IC_BOLTREE

Here we look at an example of a custom tree profile: ZBD_BUSCTR. The component set for this tree profile is IUICALL.

Tree Profile Attributes
1) Profile Name: Technical name of the tree profile.
2) Description: Meaningful description for the tree profile
3) Component Set: The component set loaded at run time; the components in the hierarchy of the tree profile should belong to this component set.
4) Rows per page: The default value for rows per page is 16 if undefined; else if the given number in this field is reached on the profile, then paging links appear.
5) Visible Rows in Scroll Area:
   Default entry 0 means that there is no maximum height defined-> the tree will grow until “Rows per page” is reached -> no vertical scrollbar will appear;
An entry >0 defines the number

6) Change Columns: This is a flag to set the cell contents of the tree as editable.

Definition of Columns

The screenshot here shows that there are 4 columns defined, each of them having a unique column ID with a title that should be displayed in the output.

We can define up to 20 columns.

Width in % is the column width. The visible columns shouldn’t exceed 100%, if nothing is defined the system will calculate the width proportionally.

If there is a requirement to hide one of the columns, it can be done by enabling the flag “Hidden”.

Based on this column definition, this is how the tree shall appear. Of course, we have to define the hierarchy of this tree profile, explained in the next step.
**Tree Profile Hierarchy**

The screenshot above shows that there is a hierarchical approach in the output, starting with Business Partner followed by Business Agreement, premise, POD, Contract, Device Location till Device.

These objects come in context of ISU. Let's understand the hierarchy in the configuration and relate it to the output shown above.

Let us have a look at the first two columns: Node ID, Parent Node ID.

These two nodes define the parent-child relationship in the tree profile. In the screenshot above, node ID 01 is the starting node as it does not have a parent node ID. Rest all nodes follow a parent-child relation starting with Node ID 01.

The component set for this tree profile is IUICALL as defined in the first step earlier. Hence only those objects which appear under this component set can be added in the hierarchy. Objects that have been added are shown under column "External Object Name" in the screenshot.

Where a direct relation can be determined between the node and the parent node in the hierarchy, the relation name is provided in the column "Path for Determining Target Nodes". For example, the relation name is set as "PoDIsuDevLocRel" for node 21.

On relating this configuration with the output, we arrive at this hierarchical pattern.

<table>
<thead>
<tr>
<th>Node ID</th>
<th>Parent Node ID</th>
<th>Path for target nodes</th>
<th>External object name</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-</td>
<td>BuilHeader</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>01</td>
<td>BuAg</td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>12</td>
<td>Premise</td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>06</td>
<td>PointOfDelivery</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>07</td>
<td>IsuOrderItem</td>
<td></td>
</tr>
</tbody>
</table>
The first node in this hierarchy is 01 for "BuilHeader".

Here we see that node ID 07 is the parent node for nodes 10, 11, 18 and 21 which means that all the corresponding child nodes would appear at the same level for these objects; hence under Point of Delivery, we can see Contracts and Device Locations at the same level.

**Comparison of Custom Profile with Standard Tree Profile**

This tree profile is a copy of the standard tree profile IUIC_BP for BP data environment for a customer.

**Standard profile: IUIC_BP**

Let's now try to understand the differences in the standard and the custom profile.

We see three extra nodes 18, 21 and 25 in the custom profile. These nodes were added to display the relevant ECC contracts, Device Location and Device in the output as they were not available in the standard.
Node Details and its Integration with Custom Development

Let’s now look at the details of a node in the tree profile. We arrive at it by double-clicking on any one of the rows in the hierarchy. Here we take example for Node ID 07.

Node attributes

1) Sequence No. : The sequence field defines a sequence of nodes in the tree hierarchy. These nodes all have the same parent node. If no values are maintained, then these nodes appear in the same order as you entered them in the tree hierarchy. It defines the sequence of nodes at a particular level.

2) Hidden: If the flag is checked, then this node is not displayed and is relevant only as an intermediate node to derive the children under.

3) Node is Folder: This node has a folder icon and it inherits the BOL object type from its parent. It is used to define e.g. semantic or systematic arranging of objects. It determines whether or not the current node is to be defined as a folder in the tree hierarchy. If yes, all lower-level nodes are added to this folder and retain their dependencies.

4) Is Recursive: The recursive field defines whether the node calls itself recursively.

5) Maximum Sibling nodes: This field defines the maximum number of siblings that the current node can possess, in order to be displayed as expanded when you call the tree hierarchy

6) Maximum Child nodes: This field defines the maximum number of child nodes that the current node can possess, in order to be displayed as expanded when you call the tree hierarchy

7) Presentation ID: This field defines which attributes and symbols should be displayed for this node. The attributes belong to the external object name already defined for this node. We can also display fields that are directly available via a relation from the external object. This shall be covered in the next section.

In addition to these attributes, there are four other flags available on each tree node which define the behavior of the node display. This is where the custom development can be hooked in the configuration.
Custom Development Associated with BOL Tree

In addition to these attributes shown above on a node, there are four other flags available on each tree node which define the behavior of the node display. This is where the custom development can be hooked in the configuration.

With every node in the tree profile, there exists an option for defining a user class exit. In the example taken, the standard tree profile has the standard class CL_CRM_IU_BT_ENH_BP defined against many of the nodes. This class implements the interface "IF_IU_IC_BOL_TREE".

![Class Builder: Display Class CL_CRM_IU_BT_ENH_BP](image)

In order to enhance this class with custom requirements, you can inherit it with a custom class that can be maintained with the relevant nodes in the tree profiles. The standard tree profile has the class CL_CRM_IU_BT_ENH_BP and the custom tree profile has the custom class ZCL_CRM_IU_BT_ENH_BP.

![Display View "Hierarchy": Overview](image)
Let's try to understand how this class may be used and how they relate to the flags in the configuration.

1) **Change relation**: If the existing relation for the node is to be over-written using an exit, then this flag is checked.

In the custom tree profile, the child nodes pertaining to object IsuDeviceLocation are automatically derived from the relation PoDIsuDevLocRel with the parent node "PointOfDelivery". Hence this flag was not checked. But in situations, where it needs to be over-ridden or a custom logic added to derive the child nodes, then this flag should be checked. If this flag is checked, then the method IF_IU_IC_BOL_TREE~GET_CHILDREN gets invoked if implemented.

2) **Change Presentation**: Say, there is a requirement to display a derived value or status symbol, for example, in one of the columns for the node. This column is either undefined in the presentation ID for that node or should be over-ridden. In order to do this, check this flag. It invokes the method IF_IU_IC_BOL_TREE~GET_ATTRIBUTE in the class if implemented.

3) **Change Visibility**: This flag determines if the current node should be hidden during run time. If this flag is checked, the method IF_IU_IC_BOL_TREE~IS_HIDDEN triggers if implemented.

4) **Re-render cell**: In order to change the appearance of field on the tree in the UI, this flag needs to be checked. For example, a requirement to display Yes/No values for a column as checkboxes marked/unmarked needs to have this flag marked. There are other options also available, like displaying of radio-buttons, icons, or open a link by setting a javascript code.

In all these methods, the functionality can be set for a particular node/column by applying appropriate filters in the code logic. All these methods have either the node or column name/key or both as an importing parameter in their signature which can act as filters so that the required functionality is available in the respective nodes/columns.

There is an additional method IF_IU_IC_BOL_TREE~REFRESH in the class which is generally used to clear the global internal tables if required.

### Configuration of Presentation IDs

This is a representation of the columns for a particular node in the tree profile. Custom presentation IDs can be created to suit custom requirements and associated with the node in the tree profile nodes in the hierarchy.

Let's take example of presentation node ZBUS_PREM for object Premise.
There are 3 columns defined for ZBUS_PREM. For every column, the column values maintained in "Value Path A", "Value Path B" and "Value Path C" are concatenated and displayed. So, for example, if there are three different attributes of the object Premise which need to be displayed in column 1, then BOL value paths are maintained in each of the value paths starting from BOL object type defined in External Object Name. The value paths may also contain attributes that can be derived with a direct relation to the Premise object as seen here in column ID 2. It has "Value path A" as "/PremiseConnObjRel//SHORT_WITH_STREET". A maximum of three value paths can be combined.

The syntax for the path is the following:

"//<BOL_Atrribute_Name>" or
//<BOL_Relation_Name1>/<BOL_Relation_Name2>/.../<BOL_Atrribute_Name>". The value of the attribute with the name "BOL_Atrribute_Name" will be displayed in the cell.

The relations in the path can be used optionally to display attributes of objects which are defined beneath the current object in the GenIL model.

Texts from Online Text Repository (OTR) can also be defined in the value paths; for example column ID 1 has "Value path A" as "OTR//CRM_IST_SO/PRM".

Icons can also be associated with the presentation node as shown above. The flag "Is Link/Event ID" if marked; creates a hyper link on the column data. A custom event can also be associated with this hyperlink by giving an event name. In standard, if only the flag is checked without the event name, then the default event name is "GENERIC" which opens the details of the relevant object.
Here is the screenshot of this node as it appears in the UI.

The icon in the first column comes from the configuration of the presentation ID as explained above. The hyper link for Premise comes from the flag "Is Link/Event ID". Clicking on "Premise" navigates to the technical details of the premise.

A cell can be made editable if the flag "Editable/Path" is marked.

If the **Edit Mode** is active and the flag is marked the value defined in Edit Value Path is rendered as an input field. The value paths A, B and C are not relevant in edit mode.

### One Click Actions

One click actions can be used to trigger events or launch transactions. One of the OCA implemented in this tree profile is for the Premise node. This appears in the column "Additional Information" on the presentation ID ZBUS_PREM for the node Premise. The text displayed in the output is "ECC Premise Environment".

One click Action ID is the unique name for this OCA. An event handler method of the parent UI component is called when an OCA is selected; this key is passed to differentiate between the OCAs. The parent UI component is the one where the generic ISU BOL tree component IUBOtree will be used.
1) Alternative Column ID is the column no. on which the OCA should be placed. Default is column 1.
2) Sequence no.: Defines the sequence of the OCAs
3) An icon can be linked to OCA in field "Icon Name".
4) Text to be displayed is filled in field "Text" as above.
5) One Click Action Tooltip is the tooltip for the OCA icon and text

Assign tree Profiles to Overview

This configuration is optional, but a good practice. Here the tree profiles for this example are assigned to an overview profile.

Transaction Code: CRM_ISU_IMG,

Or, this is also available in SPRO under the path.

SAP Customizing Implementation Guide->Customer Relationship Management->Industry-Specific Solutions->Utility Industries->Settings for User Interfaces->Settings for Object Workbench->Assign Profiles to Overview
Tree profiles are assigned to the overview profile. Here, in our example, the relevant tree profile is the BP tree profile ZBD_BUSCTR assigned under overview profile ZBUS_OVW.

**Display View "Maintenance view for CRMC_IU_IC_OVW": Details**

<table>
<thead>
<tr>
<th>Overview Profile</th>
<th>ZBUS_OVW</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintenance view for CRMC_IU_IC_OVW</td>
<td></td>
</tr>
<tr>
<td><strong>BP Tree Profile</strong></td>
<td>ZBD_BUSCTR</td>
</tr>
<tr>
<td><strong>SL Tree Profile</strong></td>
<td>ZPML_SL2</td>
</tr>
<tr>
<td><strong>TD Tree Profile</strong></td>
<td>ZFS_TD</td>
</tr>
</tbody>
</table>

**Implementation of Generic Tree Component IUBOTree**

After completing the configuration of the BOL tree along with custom requirements, we proceed to the implementation details.

**Enhancement in Parent Component**

First of all define in the parent component a usage of the IUBOTREE component if not defined.

In this example, the parent component is IUICOVW where a component usage for IUBOTree is already defined.

The navigation links should be set up between the parent and the BOL tree component and the delegation to outbound plugs done.

The next step is to set the tree profile which should be used in the tree component. This shall be done in the method WD_USAGE_INITIALIZE of the parent component controller class.

Then depending on what object type is defined as root object in the tree profile the “ROOTOBJECTS” node should be filled with root objects. This can be done via binding a node of the parent component to the “ROOTOBJECTS” node or via filling the “ROOTOBJECTS” node-collection with objects manually. In the
example taken, BuildHeader is the starting root node of the tree profile. Hence, the context node "ACCOUNT" in the component controller is populated during the binding. The tree profile name is picked up from the configuration maintained in transaction CRM_ISU_IMG. In case this configuration is not maintained, you can hard code the tree profile name in the method WD_USAGEInicialize.

### Tree Properties

The behavior of the tree can be influenced by setting the tree properties in the parent component. This can be done in the same way as setting the tree profile in the WD_USAGEInicialize.

### Event handling in tree

It is possible to define 3 types of events in the tree:

- **Link Events (in each cell)**
  
  To define a link event cell attribute "Is Link" is marked and Link ID is defined in presentation node.

- **One Click Action Events (OCA Events)**
  
  To define an OCA event OCA with OCA-ID is defined in the node hierarchy.

- **Toolbar Button Events**
  
  Toolbar buttons cannot be defined via customizing, but via implementation.

To handle the events in the parent component the tree provides an event service. This event_service is a public component node of the tree component. The event_service node implements the interface 

IF_CRM_IU_BOL_TREE_EVT_SERVICE.

This is how to subscribe to the tree events in the method WD_USAGEInicialize of the parent component controller class.
The component controller class should be enhanced to implement the interface IF_CRM_IU_BOL_TREE_EVT_HANDLER if not already done by standard.

Sample code

```
Sample code
    tree_event_service =
    cl_iubotree_bspwdcomponen_cn08=>get_event_service(
        ir_parent_component = me iv_tree_usage_name = iv_usage->usage_name ).
    IF tree_event_service IS BOUND.
       tree_event_service->subscribe( event_handler = me event_type = 'OCA' event_id = space ).
       tree_event_service->subscribe( event_handler = me event_type = 'BUTT' event_id = space ).
       tree_event_service->subscribe( event_handler = me event_type = 'LINK' event_id = space ).
    ENDIF.
```

Now whenever an event is triggered (link, OCA or button is clicked) the method
IF_CRM_IU_BOL_TREE_EVT_HANDLER~HANDLE_EVENT of your subscribed handler is called.
OCA and links

The method IF_CRM_IU_BOL_TREE_EVT_HANDLER~IS_EVENT_ACTIVE is called for links and for OCAs before they are displayed, so you can control whether the actions/links which are defined in customizing should be active.

Sample code to trigger transaction launcher on a one click action, in our example, the ECC Premise Environment.

```abap
METHOD if_crm_iu_bol_tree_evt_handler~handle_event.
  DATA : lr_navigation TYPE REF TO if_crm_ui_navigation_service,
  lr_coll TYPE REF TO if_bol_bo_col,
  lr_entity TYPE REF TO cl_crm_bol_entity.

  CALL METHOD super->if_crm_iu_bol_tree_evt_handler~handle_event
    EXPORTING
      event_type = event_type
    event_id = event_id
    event_entity = event_entity
    node_id = node_id.

  lr_navigation = cl_crm_ui_navigation_service=>get_instance( me ),
  IF lr_navigation IS BOUND.
    CASE event_id.
      '& Premise Environment
      WHEN 'Z_ECC_PREM'.
        CREATE OBJECT lr_coll TYPE cl_crm_bol_bo_col.
        lr_coll->add( event_entity ).
        lr_navigation->navigate( iv_link_id = 'ZLL_PREENV' iv_data_collection = lr_coll ).
      END_CASE.
    ENDIF.
  ENDMETHOD.
```

Here based on the one click action ID 'Z_ECC_PREM', the transaction launcher for Premise is called via logical link 'ZLL_PREENV' defined in configuration.

Toolbar buttons

In the method IF_CRM_IU_BOL_TREE_EVT_HANDLER~ADJUST_TOOLBAR_ACTIONS it is possible to adjust the toolbar buttons. It is possible to adjust (remove, sort, deactivate) the tree own buttons (like position, next, expand all, collapse all, and it's possible to define your own buttons:

```
METHOD if_crm_iu_bol_tree_evt_handler~adjust_toolbar_actions.
  DATA : ls_button TYPE crmt_thtmlb_button.
  ls_button-id = 'BUTTON_1'.
  ls_button-text = 'Button 1'.
  ls_button-tooltip = 'Click Button 1'.
  ls_button-on_click = 'BUTTON_1'.
  ls_button-on_client_click = ''.
  ls_button-enabled = abap_true.
  APPEND ls_button TO ct_toolbar_actions.
ENDMETHOD.
```
Related Content

www.sdn.sap.com

Technical Data Environment

SAP Community Network Forums » Industries » IS Utilities

For more information, visit the Customer Relationship Management homepage.
Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.