



## CRM 5.0 IC WEBCLIENT INBOX BUSINESS TRANSACTION SEARCH

Document Information		
Version:	1.1	
Date:	24/03/2008	
Author:	Johan van Zijl	
Document History		
Version (starting with 1.0)	Status (Draft/Review/Final)	Date (DD.MM.YY)
1.0	Draft	19.03.2008
1.1	Final	24.03.2008

## TABLE OF CONTENTS

<b>1</b>	<b>OVERVIEW .....</b>	<b>1</b>
1.1	Definition .....	1
1.2	Benefits .....	1
1.3	Steps .....	1
1.4	Enhancement Scenario .....	2
1.5	Caveats .....	2
<b>2</b>	<b>CUSTOMIZING.....</b>	<b>3</b>
2.1	Definition of the Index .....	3
2.2	Flat Types.....	4
2.3	Table Types.....	5
2.4	Activating the Index.....	5
<b>3</b>	<b>INITIAL STEPS .....</b>	<b>6</b>
3.1	Overview .....	6
3.2	Database Table .....	6
3.3	Initial Load Report.....	7
3.3.1	Overview.....	7
3.3.2	<i>Some Notes on the Standard Report.....</i>	<i>7</i>
3.3.3	<i>Custom Load Report.....</i>	<i>8</i>
3.4	Badi Implementation .....	9
3.5	Determining the Index .....	10
<b>4</b>	<b>CLASS ZCL_CSC_SOS_UPDATE.....</b>	<b>11</b>
4.1	Overview .....	11
4.2	Get_instance .....	11
4.3	Badi_change_before_update .....	12
4.4	Populate_dates .....	15
4.5	Get_date.....	16
4.6	Populate_partners.....	16
4.7	Get_bp.....	19
4.8	Populate_status.....	20
4.9	Map_Native_States.....	21
<b>5</b>	<b>CRM_IC_INBOX_BADI.....</b>	<b>22</b>
5.1	Overview .....	22
5.2	Before_Search .....	23
5.3	After_search .....	24
5.4	Custom_hit_list_sort.....	25
5.5	Zzmap_sort_fields.....	25
5.6	ZCSC_INBOX_PARAMETERS .....	27

## 1 OVERVIEW

### 1.1 Definition

This document describes the enhancement of the IC WebClient using the Business Transaction Search in CRM 5.0. The Business Transaction Search (BT Search) allows you to define an index table that can be used as an alternative to the standard reporting framework.

### 1.2 Benefits

The BT Search allows you to overcome a number of common problems encountered in the Inbox, namely:

- **Poor performance**
- **Custom Search Fields**
- **Inconsistent Search Results**

The typical IC WebClient Project deals with all 3 of the above problems. Poor performance is almost inherent to the standard Inbox as the database structure of the One Order Framework is extremely flexible, but complex.

If you add custom search fields to the Inbox, you must either implement filters in the AFTER\_SEARCH method of BADI CRM\_IC\_INBOX\_BADI or you must implement BT Search. Using filters is always a bad idea, as performance suffers and they lead to unexpected results.

Finally, the Inbox is used to select Interactions, Service Tickets, Follow Ups, Sales Orders, etc. All of these transactions have different customizing settings and it is sometimes impossible to align of them in the standard Search Results. BT Search allows you to do this.

### 1.3 Steps

The BT Search allows you to define an Index Table in the IMG. The IMG transaction will generate the following components for you:

- A Database Table that will be your “*Index*”
- An initial load report for populating the Database Table
- A Badi Implementation (ORDER\_SAVE) used to populate the delta entries to the Index when a document is saved.

However, you will have to modify the generated Report and Badi in order to achieve the desired results.

You must also create an Implementation of Badi CRM\_IC\_SOS\_INDEX in order to determine the Index to be used.

An implementation of Badi CRM\_IC\_INBOX\_BADI will also be required for doing some fancy footwork.

The rest of the document will describe the required steps in more detail.

## 1.4 Enhancement Scenario

The enhancement described in this document caters for the following requirements:

- An Index only for Service Tickets and Follow ups
- Multiple Customer Specific Search Fields
- Customer Specific Search Fields

## 1.5 Caveats

BT Search completely replaces the standard search. It's a lot of work as you really end up rebuilding the entire search logic of the Inbox. The BT Search is not a simple enhancement. It needs a lot of tweaking. This document assumes that the reader has an understanding of One Order documents, their customizing and **ABAP knowledge**.

**This document is only applicable to CRM 5.0.** The BT Search functionality has changed significantly in CRM 2007 and many of the problems and solutions described here will hopefully not be encountered. I have not verified this beyond the point of looking at the customizing tables in CRM 2007.

***This document does not deal with adding additional search parameters to the UI or the BOL. That is a separate topic on its own. Here we assume it has already been done.***

***I have not tested this Index with an Inbox containing Workflow Items or E-mails as well. Some code in this document may be incompatible with Workflow/E-mail Items, but I just don't know.***

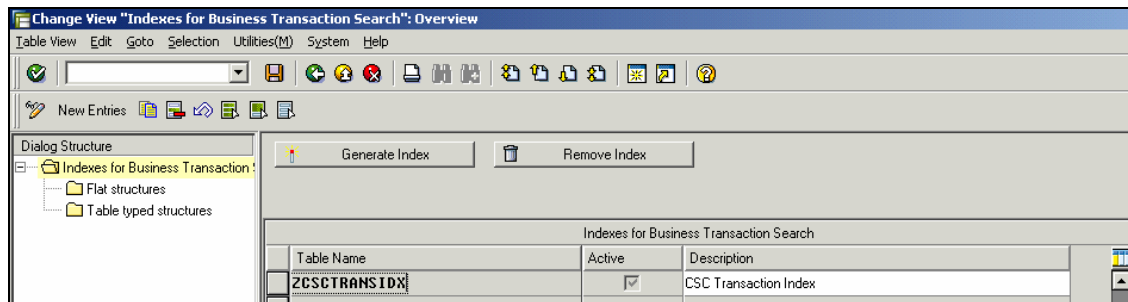
## 2 CUSTOMIZING

### 2.1 Definition of the Index

The first step involves defining the structure of the Index table in the IMG (Transaction SPRO). The menu path is:

**IMG: Customer Relationship Management → Interaction Center WebClient → Customer-Specific System Modifications → Business Transaction Search → Define Search Index for Business Transaction Search**

The first step is to define the name of the Transaction Index. It must start with a Z or Y.



Two types of fields can be included in the Index, namely:

- Flat Types – Such as Orderadm\_h, Activity\_h, etc
- Table Types – Such as Partner, Orderadm\_i, etc

Unfortunately, the developer of this module thought Status and Appointment is also flat types, so we will have to fix this later.

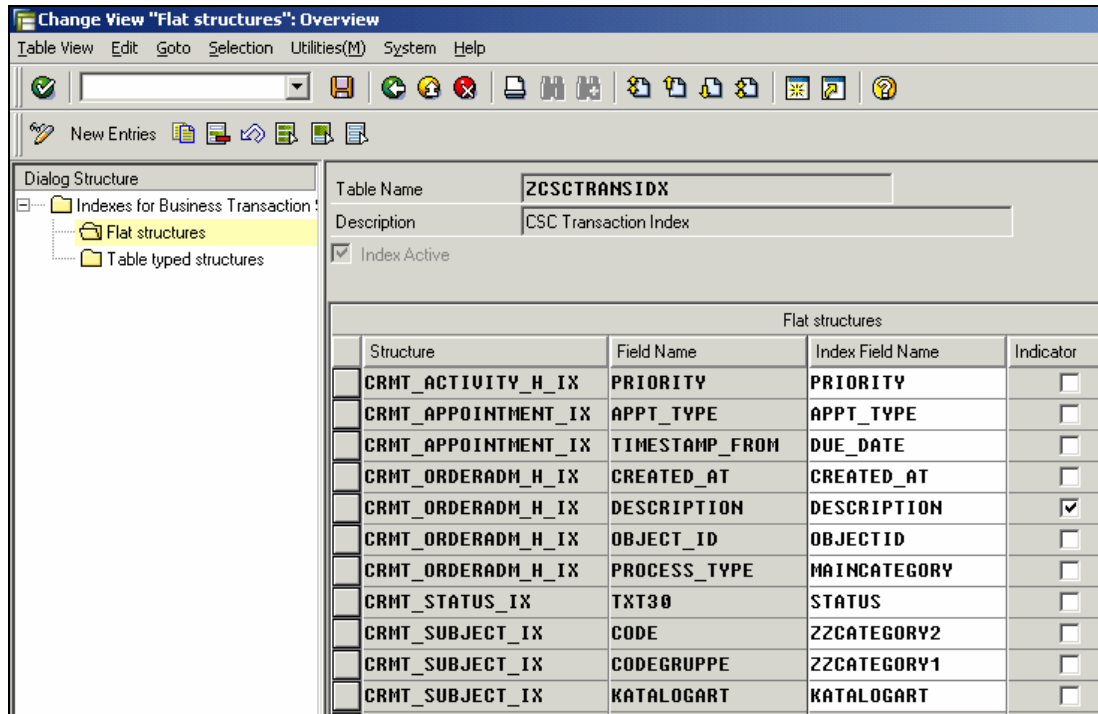
#### TIP:

Try to keep the names of your index fields the same as the names of the Standard Inbox Search Parameters.

E.g. use MAINCATEGORY instead of PROCESS\_TYPE as field name

## 2.2 Flat Types

Below is a screenshot of my flat type definitions.



Change View "Flat structures": Overview

Table Name: ZCSCTRANSIDX  
 Description: CSC Transaction Index  
 Index Active

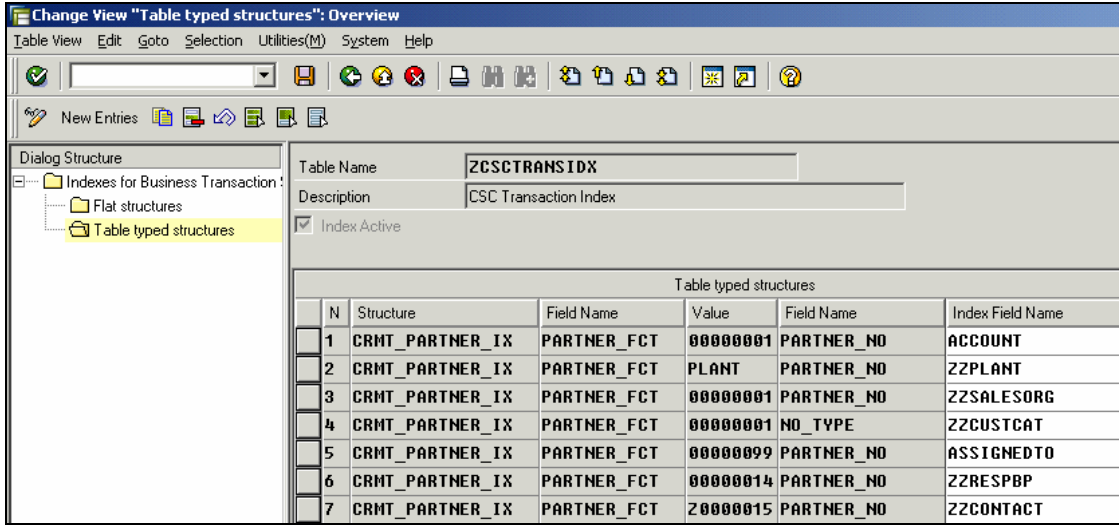
Flat structures			
Structure	Field Name	Index Field Name	Indicator
CRMT_ACTIVITY_H_IX	PRIORITY	PRIORITY	<input type="checkbox"/>
CRMT_APPOINTMENT_IX	APPT_TYPE	APPT_TYPE	<input type="checkbox"/>
CRMT_APPOINTMENT_IX	TIMESTAMP_FROM	DUE_DATE	<input type="checkbox"/>
CRMT_ORDERADM_H_IX	CREATED_AT	CREATED_AT	<input type="checkbox"/>
CRMT_ORDERADM_H_IX	DESCRIPTION	DESCRIPTION	<input checked="" type="checkbox"/>
CRMT_ORDERADM_H_IX	OBJECT_ID	OBJECTID	<input type="checkbox"/>
CRMT_ORDERADM_H_IX	PROCESS_TYPE	MAINCATEGORY	<input type="checkbox"/>
CRMT_STATUS_IX	TXT30	STATUS	<input type="checkbox"/>
CRMT_SUBJECT_IX	CODE	ZZCATEGORY2	<input type="checkbox"/>
CRMT_SUBJECT_IX	CODEGRUPPE	ZZCATEGORY1	<input type="checkbox"/>
CRMT_SUBJECT_IX	KATALOGART	KATALOGART	<input type="checkbox"/>

Important things to note:

- The Indicator check box allows you to switch on Wildcard searches for certain fields
- I had to populate Status Profile and User Status into a single field for a reason to be explained later. I used TXT30 in order to get space on my index (but we don't really update TXT30 to the Index)
- Always use the timestamps for CREATED\_AT and DUE\_DATE otherwise you crash some standard Inbox functionality

## 2.3 Table Types

Below my table type parameters:

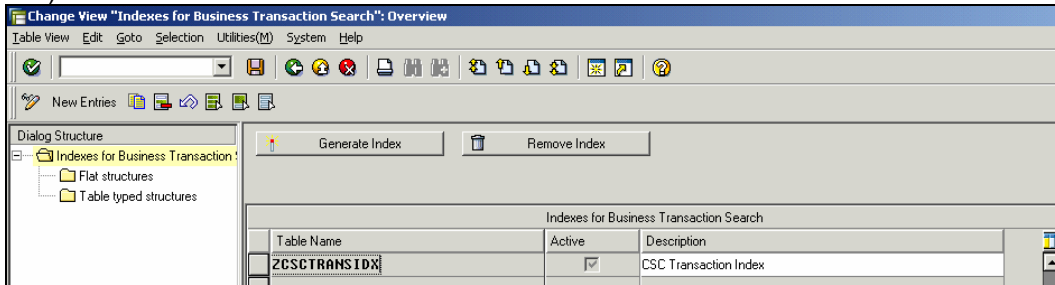


Important things to note:

- The NO\_TYPE field for ZZCUSTCAT is just another placeholder, we will populate with other data.
- The specification of Partner Functions is pretty much useless. On an Activity Transaction your ACCOUNT is stored in partner function 00000009(Activity Partner) and on a Service Ticket partner function 00000001(Sold-to Party) is used. So once again we have to do some fixing of these later on. This counts for most of the other partner functions as well.

## 2.4 Activating the Index

The last step is to activate the Index on the first screen. You have to select the Index and Click the “Generate Index” button (If your index already exists you have to “Remove Index” first).



Generating the Index will do the following:

- Generate a database table named: **ZCSCTRANSIDX**
- Generate a Report named: **ZCSCTRANSIDX\_ILOAD**
- Generate a Badi Implementation named: **ZCSCTRANSIDX\_BADI**

This is where the customizing stops and the coding starts.

### 3 INITIAL STEPS

#### 3.1 Overview

Each time you Delete and Generate your Index in the IMG, all the generated objects are deleted and recreated. **Any code changes you applied to these objects are lost!**

However, the **generated code must be changed**, so I moved most of my code to my own ABAP Class called **ZCL\_CSC\_SOS\_UPDATE**.

#### 3.2 Database Table

I had no need to change the generated database table. The system automatically adds Client and GUID as key fields to the generated table (no need to include them in the IMG).

After each generation in the IMG, you must run the initial load program again to populate the table again.

My table definition is shown below:

Field	Key	Initi...	Data element	Data Type	Length	Decim...	Short Description
<u>CLIENT</u>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MANDT	CLNT	3	0	Client
<u>GUID</u>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CRMT_OBJECT_GUID	RAW	16	0	GUID of a CRM Order Object
<u>PRIORITY</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PRIORITY	NUMC	1	0	Activity Priority
<u>APPT_TYPE</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_APPTYPE	CHAR	12	0	Technical Name for Date Type
<u>DUE_DATE</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_DATE_TIMEST	DEC	15	0	Time Stamp (Date and Time) of a Date/Duration
<u>CREATED_AT</u>	<input type="checkbox"/>	<input type="checkbox"/>	COMT_CREATED_AT	DEC	15	0	Created At (Output in User Time Zone)
<u>DESCRIPTION</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PROCESS_DES	CHAR	40	0	Transaction Description
<u>OBJECTID</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_OBJECT_ID_D	CHAR	10	0	Transaction Number
<u>MAINCATEGORY</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PROCESS_TYP	CHAR	4	0	Business Transaction Type
<u>STATUS</u>	<input type="checkbox"/>	<input type="checkbox"/>	J_TXT30	CHAR	30	0	Object status
<u>ZZCATEGORY2</u>	<input type="checkbox"/>	<input type="checkbox"/>	QCODE	CHAR	4	0	Code
<u>ZZCATEGORY1</u>	<input type="checkbox"/>	<input type="checkbox"/>	QCODEGRP	CHAR	8	0	Code Group
<u>KATALOGART</u>	<input type="checkbox"/>	<input type="checkbox"/>	COMT_CATALOG	CHAR	2	0	Catalog
<u>ACCOUNT</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	32	0	Partner Number
<u>ZZPLANT</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	32	0	Partner Number
<u>ZZSALESORG</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	32	0	Partner Number
<u>ZZCUSTCAT</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	2	0	Partner Number Type (e.g. GUID, Pers. No., ...). Internal
<u>ASSIGNEDTO</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	32	0	Partner Number
<u>ZZRESPBP</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	32	0	Partner Number
<u>ZZCONTACT</u>	<input type="checkbox"/>	<input type="checkbox"/>	CRMT_PARTNER_NO	CHAR	32	0	Partner Number

## 3.3 Initial Load Report

### 3.3.1 Overview

The initial load report is used to populate existing business transactions **into the index table**.

I copied the generated load program from **ZCSCTRANSIDX\_ILOAD** to **ZCSCTRANSIDX\_ILOAD\_CUSTOM**.

This will allow you to make changes to the Index and the Initial Load Program at the same time without losing your code each time you regenerate the Database table.

### 3.3.2 Some Notes on the Standard Report

Listing the code of the standard will take up too much space. But I want to draw your attention to certain features and limitations (also found in the Badi).

- There is no mechanism to delete all entries in the Database Table. This may become necessary in downstream systems.
- In my case we did not need Interaction Records in the Inbox. The standard report has no ability to filter or exclude certain document types from the Index. Excluding Interaction Records from the Index has a huge performance advantage.
- The Standard Report and Badi make a call to a Badi **CRM\_IC\_SOS\_PARAM**. I found this Badi to be useless and removed the calls to it completely in my code.
- A CRM Document can have multiple Appointments (Dates) and Statuses active at the same time. However, these are defined as flat types. The standard code contains the following gems:
  - `READ TABLE lt_appointment INTO ls_appointment INDEX 1.`
  - `READ TABLE lt_status INTO ls_status INDEX 1.`

The above code means that the first and best status or date is taken and updated to the Index. This is completely incorrect.

Also, **partners, appointments and status** are relevant for the Header as well as the line Item of the document. The standard code never checks for this.

- Also, note that the **ls\_** structures (e.g. `ls_appointment`, `ls_status`, etc.) are never cleared or `sy-subrc` is never checked after the read table statements. This means, if a document contains no appointments, **the previous document's information will be updated into the index**.

The bottom line is that the generated code will never work or update the Index reliably, unless your requirements are extremely simple.

### 3.3.3 Custom Load Report

Below is my final code of the Custom Load Report. I used an iterative process of removing progressively more and more generated code until I was left with the code below.

The following are the important changes note in this program:

- Used Select-options rather than parameters for Selection Screen.
- Added code to delete all entries in the table.
- Replaced all the code with a call to method **BADI\_CHANGE\_BEFORE\_UPDATE** of class **ZCL\_CSC\_SOS\_UPDATE**. This is the same method called from the Badi.

```

*&-----*
*& Report  ZCSCTRANSIDX_ILOAD
*& Initial Load Program for Business Transaction Search
*&
*& This is a copy of generated Loader ZCSCTRANSIDX_ILOAD
*&
*&-----*
report  zcsctransidx_iloadd_custom.

tables: crmd_orderadm_h.
data: lv_entries  type i,
      lv_guid    type crmt_object_guid,
      lt_guid    type crmt_object_guid_tab.

data: lr_mapper type ref to zcl_csc_sos_update,
      lv_success type flag.

*****SELECTION SCREEN*****
selection-screen begin of block b01 with frame title text-001.
selection-screen begin of line.
selection-screen comment 1(32) warn01.
selection-screen end of line.
selection-screen begin of line.
selection-screen comment 1(32) warn02.
selection-screen end of line.
selection-screen end of block b01.

selection-screen begin of block b02 with frame.
select-options: p_patype   for crmd_orderadm_h-process_type,
                p_post    for crmd_orderadm_h-posting_date.

parameters: p_delete type flag default space.
selection-screen end of block b02.
*****END OF SELECTION SCREEN*****

*****
initialization.
*****
warn01 = '1. Ensure your first run deletes all entries!'.
warn02 = '2. Ensure you update ZFOL last!'.

```

```

*****
start-of-selection.
*****

clear: lv_entries.

lr_mapper = zcl_csc_sos_update=>get_instance( ).

if p_delete = 'X'.
    delete from zscctransidx client specified
        where client = sy-mandt.
    commit work and wait.
endif.

select guid into table lt_guid
    from crmd_orderadm_h
    where process_type in p_ptype
        and posting_date in p_post.

loop at lt_guid into lv_guid.
    lv_success =
        lr_mapper->badi_change_before_update( iv_guid = lv_guid ).
    if lv_success = 'X'.
        lv_entries = lv_entries + 1.
    endif.
endloop.

write: / lv_entries,
        'entries in Index table ZCSCTRANSIDX created.'.

```

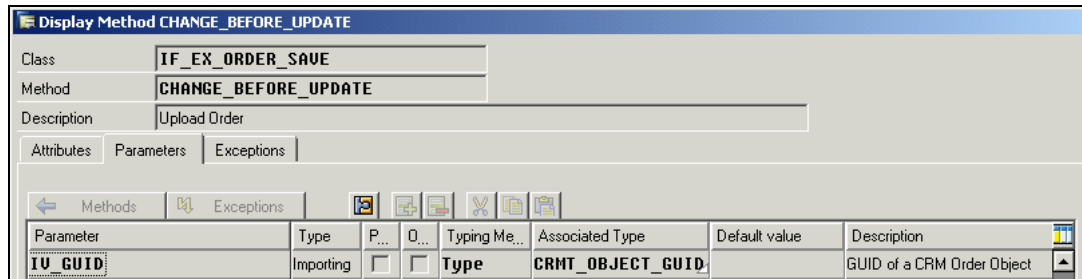
### 3.4 Badi Implementation

The Badi Implementation is based on definition ORDER\_SAVE. This Badi is triggered when a Business transaction is saved. All the code is implemented in the CHANGE\_BEFORE\_UPDATE method.

Implementation Name	ZCSCTRANSIDX_BADI	Active
Implementation Short Text	Business Transaction Search BAdI	
Definition name	ORDER_SAVE	
Attributes    Interface		
Interface name	IF_EX_ORDER_SAVE	
Name of implementing class:	ZCL_IM_CSCTRANSIDX_BADI	
Method	Implementatio..	Description
CHANGE_BEFORE_UPDATE	ABAP Code ▼	Upload Order ▲
PREPARE	ABAP Code ▼	Prepare Save
CHECK_BEFORE_SAVE	ABAP Code ▼	

The code in the initial load report and in the Badi is essentially the same. Therefore, you don't need to make any changes to the Badi initially. Ignore it until your initial load report is working correctly.

I did not make a copy of the generated Badi, because I did not want to risk having 2 Badi's active at the same time. I replaced all the generated code with my code directly in the generated Badi as shown below.



```
method if_ex_order_save~change_before_update.
```

```

  data: lr_mapper type ref to zcl_csc_sos_update.
  lr_mapper = zcl_csc_sos_update=>get_instance( ).
  lr_mapper->badi_change_before_update( iv_guid = iv_guid ).

```

```
endmethod.
```

### 3.5 Determining the Index

You have to create an Implementation for Badi **CRM\_IC\_SOS\_INDEX**. This implementation is used to determine which Index is used. This can be done directly in SE19 or the following IMG path: **Customer Relationship Management → Interaction Center WebClient → Customer-Specific System Modifications → Business Transaction Search → BAdI: Determination of Search Index for Business Transaction Search**

In this example, I only have one Index and it should only be used for Inbox selections. Therefore I implemented the following code in method **GET\_INDEX\_NAME**:

```

Method: ZCL_IM_CRM_IC_SOS_INDEX->IF_EX_CRM_IC_SOS_INDEX-GET_INDEX_NAME
*IV_CALLER      Importing      Type      STRING Calling Function or Class
*IV_PROFILE     Importing      Type      STRING IC WebClient Profile
*IT_PARAMETERS Importing      Type      CRMT_NAME_VALUE_PAIR_TAB Parameter Table of Name-
Value Pairs
*EV_INDEXNAME  Changing       Type      TABLE_NAME Table name
method IF_EX_CRM_IC_SOS_INDEX~GET_INDEX_NAME.

```

```

* Only use this Index for Inbox Search, Allow other searches(Irecs)
to happen via framework.

```

```

* Index below does not contain Irecs
  if iv_caller = 'CL_CRM_QUERYAUI_RUN_BTIL'.
    ev_indexname = 'ZCSCTRANSIDX'.
  endif.
endmethod.

```

#### PLEASE NOTE:

BT Search is also used for Interaction History searches. However, the requirements for this Index will be different, and a separate Index may be required. The Index explained in this document does not work for Interaction History Search.

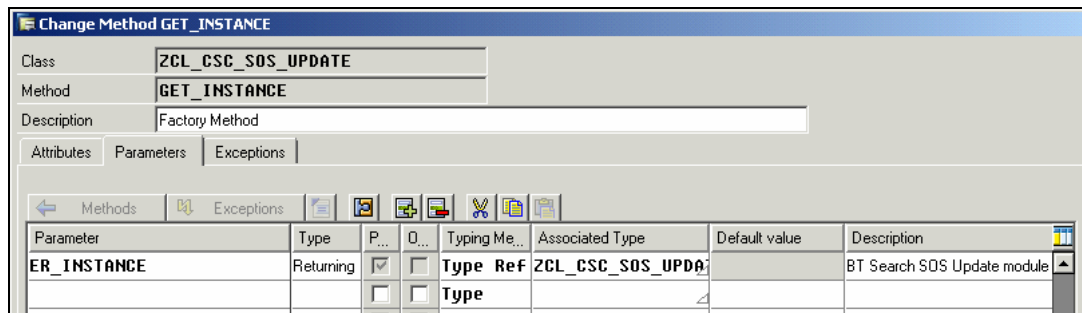
## 4 CLASS ZCL\_CSC\_SOS\_UPDATE

### 4.1 Overview

The bulk of the code required to update the Index Table has been moved to this custom class. This section describes the various methods Implemented in this class.

### 4.2 Get\_instance

Only one instance of this class will ever be required(Instance is kept in Static Attribute gr\_current), so I created a factory method that provides an instance. I do initialization of a global table to cache Inbox Status mappings in this method. This code should rather be implemented in the class or instance Constructor(more reliable).



```
method GET_INSTANCE.
```

```
    if gr_current is initial.
        create object gr_current.
```

```
    * Cache Status Mapping (table CRMC_AUI_MAP_STA)
    SELECT * FROM crmc_aui_map_sta INTO TABLE gt_status_mapping.
    SORT gt_status_mapping BY map_item_type inbox_status item_status.
    endif.
```

```
    er_instance = gr_current.
```

```
endmethod.
```

### 4.3 Badi\_change\_before\_update

This method controls the update to the Transaction Index table and is called from the Initial Load report and the Order\_Save Badi. The code of this method is based on the code in the code generated by the system.

Important to Note:

- I first call CRM\_ORDERADM\_H\_READ\_OW to read the header of the document and then call method **Filter**. This prevent an entire ORDER\_READ on documents that are not needed.
- `translate ls_newentry-description to upper case`. If you don't do this your search on Description will not work.
- I needed complex processing for partners, dates and status so I created specific methods for those:
  - `me->populate_status`
  - `me->populate_dates`
  - `me->populate_partners`

Change Method BADI_CHANGE_BEFORE_UPDATE																																																
Class		ZCL_CSC_SOS_UPDATE																																														
Method		BADI_CHANGE_BEFORE_UPDATE																																														
Description		Called by IF_EX_ORDER_SAVE~CHANGE_BEFORE_UPDATE																																														
Attributes		Parameters																																														
Parameters		Exceptions																																														
<div style="display: flex; justify-content: space-between; align-items: center;"> <span>← Methods</span> <span>⚙ Exceptions</span> <span>📄</span> <span>🔍</span> <span>📄</span> <span>📄</span> <span>📄</span> <span>📄</span> <span>📄</span> </div> <table border="1"> <thead> <tr> <th>Parameter</th> <th>Type</th> <th>P...</th> <th>O...</th> <th>Typing Me...</th> <th>Associated Type</th> <th>Default value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>IV_GUID</td> <td>Importing</td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td>Type</td> <td>CRMT_OBJECT_GUID</td> <td></td> <td>GUID of a CRM Order Object</td> </tr> <tr> <td>EV_SUCCESS</td> <td>Returning</td> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td>Type</td> <td>FLAG</td> <td></td> <td>General Flag</td> </tr> <tr> <td></td> <td></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td>Type</td> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td>Type</td> <td></td> <td></td> <td></td> </tr> </tbody> </table>									Parameter	Type	P...	O...	Typing Me...	Associated Type	Default value	Description	IV_GUID	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_OBJECT_GUID		GUID of a CRM Order Object	EV_SUCCESS	Returning	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	FLAG		General Flag			<input type="checkbox"/>	<input type="checkbox"/>	Type						<input type="checkbox"/>	<input type="checkbox"/>	Type			
Parameter	Type	P...	O...	Typing Me...	Associated Type	Default value	Description																																									
IV_GUID	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_OBJECT_GUID		GUID of a CRM Order Object																																									
EV_SUCCESS	Returning	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	FLAG		General Flag																																									
		<input type="checkbox"/>	<input type="checkbox"/>	Type																																												
		<input type="checkbox"/>	<input type="checkbox"/>	Type																																												

```

method badi_change_before_update.
* Update Business Transaction Search Table
*****
*/ Include the code below into: ZCL_IM_CSCTRANSIDX_BADI
*****
* data: lr_mapper type ref to zcl_csc_sos_update,
* lr_mapper = zcl_csc_sos_update=>get_instance( ).
* lr_mapper->BADI_CHANGE_BEFORE_UPDATE( exporting IV_GUID = iv_guid
).
*****

data: lv_guid type crmt_object_guid.
data: lt_header_guid type crmt_object_guid_tab,
      lt_requested_objects type crmt_object_name_tab,
      lt_orderadm_h type crmt_orderadm_h_wrkt,
      lt_activity_h type crmt_activity_h_wrkt,
      lt_appointment type crmt_appointment_wrkt,
      lt_partner type crmt_partner_external_wrkt,
      lt_subject type crmt_subject_wrkt,
      lt_status type crmt_status_wrkt,
      lv_log_handle type balloghndl,
      ls_orderadm_h type crmt_orderadm_h_wrk,
      ls_activity_h type crmt_activity_h_wrk,
    
```

```
ls_appointment      type crmt_appointment_wrk,
ls_partner           type crmt_partner_external_wrk,
ls_subject           type crmt_subject_wrk,
ls_newentry         type zcsctransidx,
lv_success           type flag.

clear: lt_requested_objects,
      lt_activity_h,
      lt_appointment,
      lt_orderadm_h,
      lt_status,
      lt_subject,
      lt_partner,
      lt_header_guid.

lv_guid = iv_guid.
clear ev_success.

call function 'CRM_ORDERADM_H_READ_OW'
  exporting
    iv_orderadm_h_guid      = lv_guid
  importing
    es_orderadm_h_wrk      = ls_orderadm_h
  exceptions
    admin_header_not_found = 1
    others                  = 2.

lv_success = me->filter( is_orderadm_h = ls_orderadm_h  ).
if lv_success = 'X'.

append 'ACTIVITY_H' to lt_requested_objects.
append 'APPOINTMENT' to lt_requested_objects.
append 'ORDERADM_H' to lt_requested_objects.
append 'PARTNER' to lt_requested_objects.
append 'STATUS' to lt_requested_objects.
append 'SUBJECT' to lt_requested_objects.

append lv_guid to lt_header_guid.

call function 'CRM_ORDER_READ'
  exporting
    it_header_guid      = lt_header_guid
    it_requested_objects = lt_requested_objects
  importing
    et_orderadm_h      = lt_orderadm_h
    et_activity_h      = lt_activity_h
    et_appointment     = lt_appointment
    et_partner         = lt_partner
    et_subject         = lt_subject
    et_status          = lt_status
  changing
    cv_log_handle      = lv_log_handle
  exceptions
    document_not_found = 1
    error_occurred     = 2
    document_locked    = 3
    no_change_authority = 4
    no_display_authority = 5
    no_change_allowed  = 6
    others              = 7.
```

```
clear ls_newentry.
ls_newentry-guid = lv_guid.

read table lt_activity_h into ls_activity_h index 1.
if sy-subrc = 0.
  ls_newentry-priority = ls_activity_h-priority.
endif.

read table lt_orderadm_h into ls_orderadm_h index 1.
if sy-subrc = 0.
  ls_newentry-description = ls_orderadm_h-description.
  translate ls_newentry-description to upper case.
  ls_newentry-objectid = ls_orderadm_h-object_id.
  ls_newentry-created_at = ls_orderadm_h-created_at.
  ls_newentry-maincategory = ls_orderadm_h-process_type.
endif.

read table lt_subject into ls_subject index 1.
if sy-subrc = 0.
  ls_newentry-zzcategory2 = ls_subject-code.
  ls_newentry-zzcategory1 = ls_subject-codegruppe.
  ls_newentry-katalogart = ls_subject-katalogart.
endif.

* Populate Status
me->populate_status(
  exporting iv_proctype = ls_newentry-maincategory
           it_status    = lt_status
  changing  cs_entry    = ls_newentry ).

* Populate Dates
me->populate_dates(
  exporting iv_proctype = ls_newentry-maincategory
           it_dates     = lt_appointment
  changing  cs_entry    = ls_newentry ).

* Populate Partners
me->populate_partners(
  exporting iv_proctype = ls_newentry-maincategory
           it_partner   = lt_partner
  changing  cs_entry    = ls_newentry ).

modify zscstransidx from ls_newentry.
if sy-subrc = 0.
  ev_success = 'X'.
endif.
endif.

endmethod.
```

## 4.4 Populate\_dates

This method is used to populate the DUE\_DATE and APPT\_TYPE field in the Index. The update to APPT\_TYPE is pointless as it is never used and should be removed.

Change Method POPULATE_DATES							
Class		ZCL_CSC_SOS_UPDATE					
Method		POPULATE_DATES					
Description		Map Partner Fields					
Attributes		Parameters					
Parameters		Exceptions					
<div style="display: flex; justify-content: space-between;"> <span>← Methods</span> <span>↗ Exceptions</span> </div>							
Parameter	Type	P...	O...	Typing Me...	Associated Type	Default value	Description
IV_PROCTYPE	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_PROCESS_TYP		Business Transaction Type
IT_DATES	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_APPOINTMENT		
CS_ENTRY	Changing	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	ZCSCTRANSIDX		Business Transaction Search Inc

```
method populate_dates.
```

```
  data: ls_dates type crmt_appointment_wrk.
```

```
  clear cs_entry-due_date.
```

```
  case iv_proctype.
```

```
    when 'ZFOL'.
```

```
      cs_entry-appt_type = 'ORDERPLANNED'.
```

```
    when 'ZCON'.
```

```
      cs_entry-appt_type = 'ORDERACTUAL'.
```

```
    when 'ZSST' or 'ZST1'.
```

```
      cs_entry-appt_type = 'SRV_RREADY'.
```

```
    when 'ZSSI'.
```

```
      cs_entry-appt_type = 'ORDERPLANNED'.
```

```
  endcase.
```

```
  cs_entry-due_date =
```

```
    me->get_date( iv_appt_type = cs_entry-appt_type
                  it_dates     = it_dates ).
```

```
endmethod.
```

## 4.5 Get\_date

Note that REF\_KIND is hardcoded to be equal to 'A'. This means only header dates will be retrieved.

Change Method GET_DATE							
Class		ZCL_CSC_SOS_UPDATE					
Method		GET_DATE					
Description		Map Partner Fields					
Attributes		Parameters					
Parameters		Exceptions					
<div style="display: flex; justify-content: space-between;"> <span>← Methods</span> <span>⚙ Exceptions</span> </div>							
Parameter	Type	P...	O...	Typing Me...	Associated Type	Default value	Description
IV_APPT_TYPE	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_APPTTYPE		Business Transaction Type
IT_DATES	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_APPOINTMENT		
ES_DATE	Returning	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_DATE_TIMEST		Business Transaction Search Inc

```
method get_date.
```

```

  data: ls_dates type crmt_appointment_wrk.

  clear es_date.
  read table it_dates into ls_dates
    with key appt_type = iv_appt_type
    ref_kind = 'A'.

  if sy-subrc = 0.
    if ls_dates-date_to is initial.
      es_date = ls_dates-timestamp_from.
    else.
      es_date = ls_dates-timestamp_to.
    endif.
  endif.

```

```
endmethod.
```

## 4.6 Populate\_partners

In addition to populating all the partners to the Index, this method caters to 2 special requirements namely:

- Populating the Sales Organisation from the BP to the Index
- Populating the Customer Group from the BP Sales Area data into the Index

In this case the Customer had a Business Rule that a Customer Account can only exist in 1 Sales Area at a time.

In order to determine Sales Area of the BP:

```

call function 'CRM_BUPA_FRG0010_GET_LIST'
  exporting
    iv_partner_guid = lv_partner_guid
  importing
    et_sales_areas = lt_areas.
read table lt_areas into ls_areas index 1.

```

The above is ok only because of the Business Rule and a specific customer requirement.

In order to determine the Customer Group I used the function module below.

```

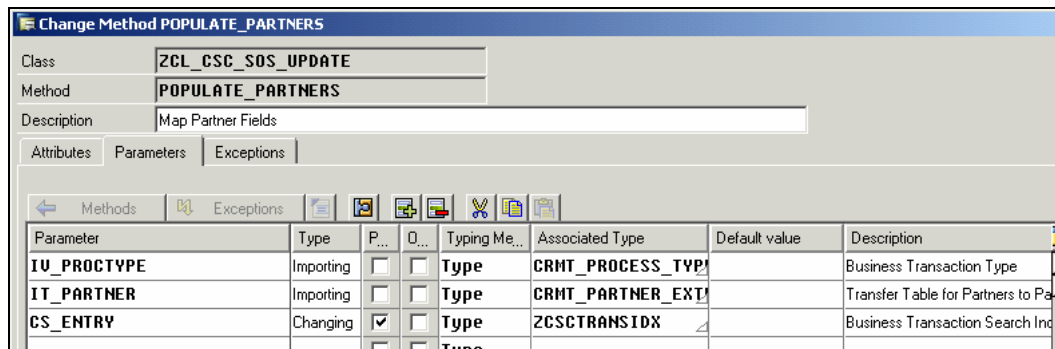
call function 'CRM_BUPA_FRG0030_GET_DETAIL'
  exporting
    lv_partner_guid = lv_partner_guid
    is_sales_area   = ls_areas
  importing
    es_data         = ls_sales.

cs_entry-zzcustcat = ls_sales-CUSTOMER_GROUP.

```

Note that in **section 2.3** we specified that Field ZCCUSTCAT should be populated from CRMT\_PARTNER\_IX-NO\_TYPE. We completely override that specification here. The only reason NO\_TYPE was used is that it is similar in type and length to field CUSTOMER\_GROUP.

The code for the full method is provided below:



The screenshot shows the 'Change Method POPULATE\_PARTNERS' dialog box. The Class is 'ZCL\_CSC\_SOS\_UPDATE' and the Method is 'POPULATE\_PARTNERS'. The Description is 'Map Partner Fields'. The Parameters tab is active, showing a table of parameters:

Parameter	Type	P...	O...	Typing Me...	Associated Type	Default value	Description
IU_PROCTYPE	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_PROCESS_TYP		Business Transaction Type
IT_PARTNER	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_PARTNER_EXT		Transfer Table for Partners to Pa
CS_ENTRY	Changing	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	ZCSCTRANSIDX		Business Transaction Search Inc

method POPULATE\_PARTNERS.

```

data: ls_partner      type crmt_partner_external_wrk,
      lv_partner_guid type bu_partner_guid,
      lt_areas        type crmt_bus_sales_area_t,
      ls_areas        type crmt_bus_sales_area,
      lv_orgunit      type objektid,
      lv_org_guid     type bu_partner_guid,
      ls_sales        type crmt_bus_set0030.

clear lv_partner_guid.
* Get Account - Try Sold-to Party First
read table it_partner into ls_partner with key partner_fct = '00000001'
                                         ref_kind = 'A'.

if sy-subrc = 0.
  cs_entry-account = me->get_bp( ls_partner-bp_partner_guid ).
  lv_partner_guid = ls_partner-bp_partner_guid.
else.
* Now try for activity partner
  read table it_partner into ls_partner with key partner_fct = '00000009'
                                         ref_kind = 'A'.

  if sy-subrc = 0.
    cs_entry-account = me->get_bp( ls_partner-bp_partner_guid ).
    lv_partner_guid = ls_partner-bp_partner_guid.
  endif.
endif.

* Get Contact Person - Z function first
read table it_partner into ls_partner with key partner_fct = 'Z0000015'
                                         ref_kind = 'A'.

if sy-subrc = 0.
  cs_entry-zzcontact = me->get_bp( ls_partner-bp_partner_guid ).
else.
* Now try for standard CP Function
  read table it_partner into ls_partner with key partner_fct = '00000015'
                                         ref_kind = 'A'.

  if sy-subrc = 0.
    cs_entry-account = me->get_bp( ls_partner-bp_partner_guid ).
  endif.
endif.

```

```
endif.

read table it_partner into ls_partner with key partner_fct = 'PLANT'
                                         ref_kind = 'A'.

if sy-subrc = 0.
  cs_entry-zzplant = ls_partner-partner_no.
endif.

* Determine the Sales Org for the BP.

call function 'CRM_BUPA_FRG0010_GET_LIST'
  exporting
    iv_partner_guid = lv_partner_guid
  importing
    et_sales_areas = lt_areas.

read table lt_areas into ls_areas index 1.
if sy-subrc = 0 and
  not ls_areas-sales_org is initial.
  cs_entry-zzsalesorg = ls_areas-sales_org.
* Determine Cust Cat *****
call function 'CRM_BUPA_FRG0030_GET_DETAIL'
  exporting
    iv_partner_guid = lv_partner_guid
    is_sales_area   = ls_areas
  importing
    es_data         = ls_sales.

  cs_entry-zzcustcat = ls_sales-CUSTOMER_GROUP.

endif.

read table it_partner into ls_partner with key partner_fct = '00000099'
                                         ref_kind = 'A'.

if sy-subrc = 0.
  cs_entry-assignedto = me->get_bp( ls_partner-bp_partner_guid ).
endif.

read table it_partner into ls_partner with key partner_fct = '00000014'
                                         ref_kind = 'A'.

if sy-subrc = 0.
  cs_entry-zzrespbp = me->get_bp( ls_partner-bp_partner_guid ).
else.
  read table it_partner into ls_partner with key partner_fct = '00000022'
                                         ref_kind = 'A'.

  if sy-subrc = 0.
    cs_entry-zzrespbp = me->get_bp( ls_partner-bp_partner_guid ).
  else.
    read table it_partner into ls_partner with key partner_fct = 'ICAGENT'
                                         ref_kind = 'A'.

    if sy-subrc = 0.
      cs_entry-zzrespbp = me->get_bp( ls_partner-bp_partner_guid ).
    endif.
  endif.
endif.

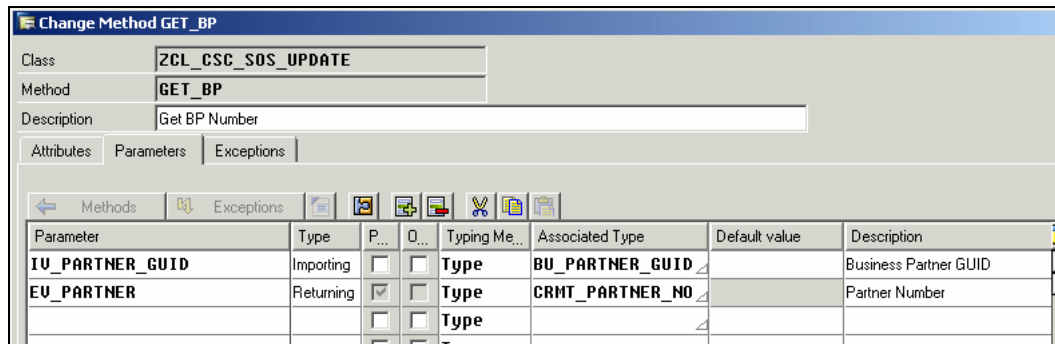
endmethod.
```

## 4.7 Get\_bp

This method is called from populate partners to provide a Business Partner Number.

Due to customizing settings the partner\_no field sometimes contain usernames or guids. This method ensures that only valid business partner numbers are populated into the index.

There may be more elegant/efficient ways of accomplishing this function.



The screenshot shows the 'Change Method GET\_BP' dialog box in SAP. The 'Class' is 'ZCL\_CSC\_SOS\_UPDATE' and the 'Method' is 'GET\_BP'. The 'Description' is 'Get BP Number'. Below the description are tabs for 'Attributes', 'Parameters', and 'Exceptions'. The 'Parameters' tab is active, showing a table with the following data:

Parameter	Type	P...	O...	Typing Me...	Associated Type	Default value	Description
IV_PARTNER_GUID	Importing	<input type="checkbox"/>	<input type="checkbox"/>	Type	BU_PARTNER_GUID		Business Partner GUID
EV_PARTNER	Returning	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	CRMT_PARTNER_NO		Partner Number
		<input type="checkbox"/>	<input type="checkbox"/>	Type			

```

method GET_BP.
  data: lv_partner type bu_partner.
  clear ev_partner.
  call function 'BUPA_NUMBERS_GET'
    exporting
      iv_partner_guid = iv_partner_guid
    importing
      ev_partner      = lv_partner.
  ev_partner = lv_partner.
endmethod.

```

## 4.8 Populate\_status

The Inbox selection will be done based on the active user status in the document. However, when you customize the status profile in the IMG, SAP assigns an internal status key. This key(CRMT\_STATUS\_WRK-STATUS) is the one the Inbox uses to select documents(full explanation to follow later).

To demonstrate the problem we face I show 2 status profiles:

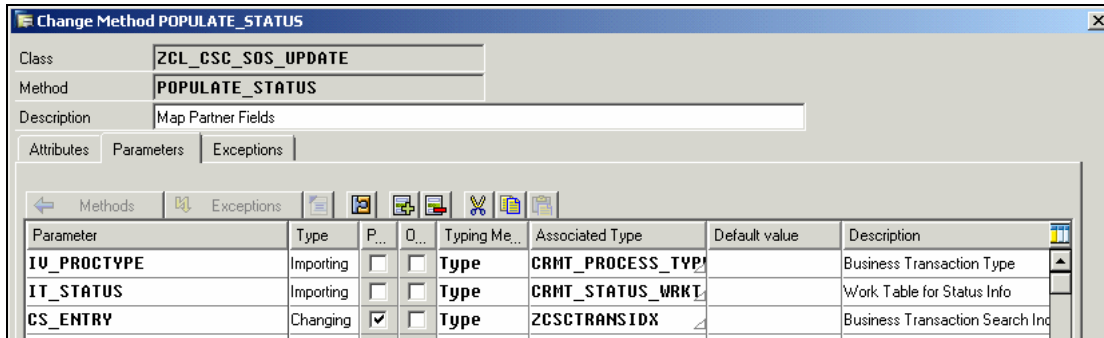
Service Ticket Status		Follow-up status	
Code	Description	Code	Description
E0001	Open	E0001	Open
E0002	In Process	E0002	Complete
E0003	Customer Action		
E0004	Complete		

Given the table above, if you were to select for “Complete” documents, the Inbox will map that to status “E0004” and “E0002”. However, that will generate a SQL Select that will return the following document to you:

- “Complete” Service Tickets(Status E0004)
- “In Process” Service Tickets(Status E0002) ← **This is incorrect!**
- “Complete” Follow-ups(Status E0002)

Therefore, we have to append User Status Procedure before updating the Index. We also, have to make some changes to the Inbox select later on.

The code is shown below:



```
method POPULATE_STATUS.
```

```

    data: ls_status TYPE CRMT_STATUS_WRK.

    loop at it_status into ls_status
      where USER_STAT_PROC ne space
        and active = 'X'.
      concatenate ls_status-USER_STAT_PROC '#' ls_status-status
        into cs_entry-status.
      condense cs_entry-status no-gaps.
    endloop.
endmethod.
```

## 4.9 Map\_Native\_States

A mapping between Inbox Status and Transaction Status has to be maintained in the IMG. This mapping table(crmc\_aui\_map\_sta) is then used by method MAP\_NATIVE\_STATES in class CL\_CRM\_AUI\_STATUS to translate Inbox Status to the relevant transaction statuses.

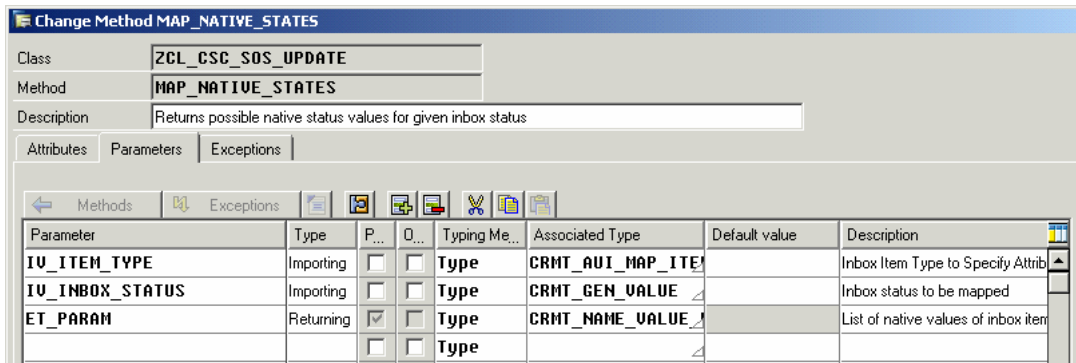
Suppose we have the following mapping table:

Service Ticket Status				
Inbox Status	Description	User Status	Status Code	Status
20	In Process	ZSERVTICK	E0002	In Process
40	Complete	ZFOLLOW	E0002	Complete
40	Complete	ZSERVTICK	E0004	Complete

The Standard MAP\_NATIVE\_STATES method will map Inbox status 40 to Transaction Status E0002 and E0004, which leads to the incorrect result described in 4.8.

Therefore, I copied the method and adjusted it to return a combination field. The new method will be therefore map Inbox Status 40 to Transaction Status ZFOLLOW#E0002 and ZSERVTICK#E0004.

This method is called from CRM\_IC\_INBOX\_BADI method BEFORE\_SEARCH. It is impossible to prevent the call of CL\_CRM\_AUI\_STATUS->MAP\_NATIVE\_STATES without modification. **Therefore, I had to change the name of the status field on the UI to ZZSTATUS, and I remap to STATUS here.**



```

METHOD MAP_NATIVE_STATES .
*Copied and modified from CL_CRM_AUI_SERVICE*****
*****
* Function returns corresponding native status values
* for given inbox status of given item type
*****
* Defintions
FIELD-SYMBOLS: <mapping> TYPE crmst_aui_map_sta.
DATA: ls_param TYPE CRMT_NAME_VALUE_PAIR,
      lv_stat_notcompl TYPE crmt_aui_notcompleted.

clear: ls_param, et_param.
ls_param-name = 'STATUS'.
* Selected Status 'Not completed'?
SELECT SINGLE stat_notcompl FROM crmc_aui_status INTO lv_stat_notcompl
WHERE inbox_status = iv_inbox_status.

*** Set status for "first selection"
IF lv_stat_notcompl EQ abap_false.
* Status NE 'Not Completed'
LOOP AT gt_status_mapping ASSIGNING <mapping>
WHERE map_item_type = iv_item_type AND inbox_status =
iv_inbox_status.

```

```

concatenate <mapping>-STATUS_PROFILE '#' <mapping>-item_status
into ls_param-value.

APPEND ls_param TO et_param.
ENDLOOP.
ELSEIF lv_stat_notcompl EQ abap_true.
* Status 'Not Completed'
LOOP AT gt_status_mapping ASSIGNING <mapping>
WHERE map_item_type = iv_item_type AND
stat_notcompl = abap_true.
concatenate <mapping>-STATUS_PROFILE '#' <mapping>-item_status
into ls_param-value.
APPEND ls_param TO et_param.
ENDLOOP.
ENDIF.

ENDMETHOD.

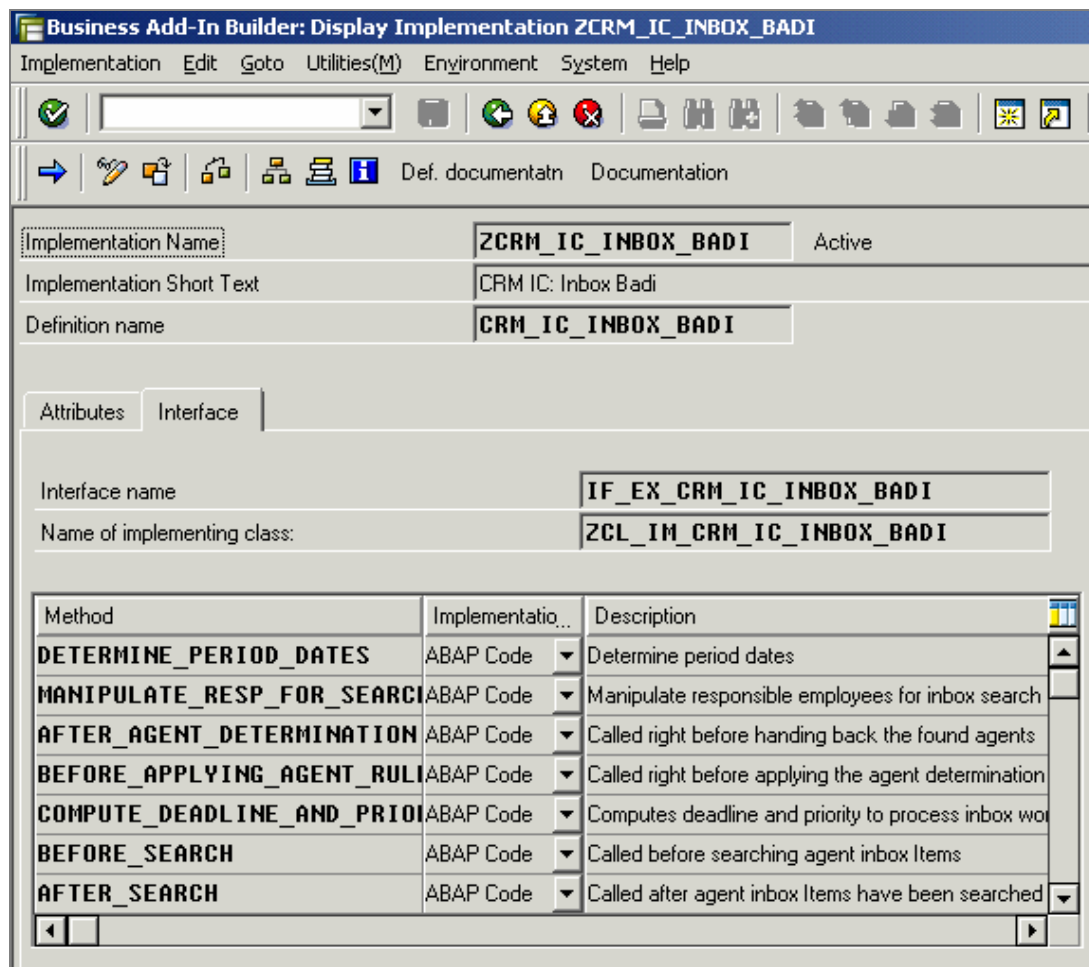
```

## 5 CRM\_IC\_INBOX\_BADI

### 5.1 Overview

This Badi allows the adjustment search parameters and results before and after the execution of the Inbox search. The methods implemented for this enhancement were:

- BEFORE\_SEARCH
- AFTER\_SEARCH
- CUSTOM\_HIT\_LIST\_SORT



The screenshot shows the 'Business Add-In Builder: Display Implementation ZCRM\_IC\_INBOX\_BADI' window. The 'Implementation Name' is ZCRM\_IC\_INBOX\_BADI, which is Active. The 'Implementation Short Text' is CRM IC: Inbox Badi. The 'Definition name' is CRM\_IC\_INBOX\_BADI. The 'Interface' tab is selected, showing the 'Interface name' as IF\_EX\_CRM\_IC\_INBOX\_BADI and the 'Name of implementing class' as ZCL\_IM\_CRM\_IC\_INBOX\_BADI. Below this, a table lists the implemented methods:

Method	Implementatio...	Description
DETERMINE_PERIOD_DATES	ABAP Code	Determine period dates
MANIPULATE_RESP_FOR_SEARCH	ABAP Code	Manipulate responsible employees for inbox search
AFTER_AGENT_DETERMINATION	ABAP Code	Called right before handing back the found agents
BEFORE_APPLYING_AGENT_RULE	ABAP Code	Called right before applying the agent determination
COMPUTE_DEADLINE_AND_PRIOR	ABAP Code	Computes deadline and priority to process inbox work
BEFORE_SEARCH	ABAP Code	Called before searching agent inbox items
AFTER_SEARCH	ABAP Code	Called after agent inbox items have been searched

## 5.2 Before\_Search

This method is used to make certain adjustments to the query parameters before the search is executed.

Important to note:

- The call to MAP\_NATIVE\_STATES uses parameter ZZSTATUS and adds new parameters with name STATUS to the query.
- For date type value 0002 must be translated to DUE\_DATE, else the underlying SQL Select is generated incorrectly.

```

method if_ex_crm_ic_inbox_badi~before_search.
  field-symbols: <fs_param> type crmt_name_value_pair,
                 <ts_param> type crmt_name_value_pair.

  data: ls_param type crmt_name_value_pair.
  data: lv_objectid type crmt_object_id,
        lv_account type bu_partner.

  data: lt_params type CRMT_NAME_VALUE_PAIR_TAB.

  data: lr_mapper type ref to zcl_csc_sos_update.

  lr_mapper = zcl_csc_sos_update=>get_instance( ).

  loop at c_query_parameters assigning <fs_param>.
    case <fs_param>-name.
      when 'ASSIGNEDTYPE'.
        if <fs_param>-value = '0001'.
          read table c_query_parameters assigning <ts_param>
            with key name = 'ASSIGNEDTO'.
          if sy-subrc = 0.
            <ts_param>-name = 'ZZRESPBP'.
          endif.
        endif.
      when 'ACCOUNT'.
        call function 'CONVERSION_EXIT_ALPHA_INPUT'
          exporting
            input = <fs_param>-value
          importing
            output = lv_account.
        <fs_param>-value = lv_account.
      when 'OBJECTID'.
        call function 'CONVERSION_EXIT_ALPHA_INPUT'
          exporting
            input = <fs_param>-value
          importing
            output = lv_objectid.

        <fs_param>-value = lv_objectid.
      when 'DATETYPE'.
        if <fs_param>-value = '0002'.
          <fs_param>-value = 'DUE_DATE'.
        endif.
      when 'ZZSTATUS'.
        lt_params = lr_mapper->map_native_states(
          iv_item_type = 'ONEORDER'
          iv_inbox_status = <fs_param>-value ).
    endcase.
  endloop.
  if not lt_params is initial.
    append lines of lt_params to c_query_parameters.
  endif.

endmethod.

```

## 5.3 After\_search

The inbox select only returns the GUIDs and standard fields of the Business Transactions found. If you added custom sort criteria or your index does not quite match standard you have to manually populate these here. In this case I overwrote all fields with the transaction index values.

```
method if_ex_crm_ic_inbox_badi~after_search.
```

*\* We only need this to populate custom search fields. For Performance done directly from BT Index*

```

field-symbols: <ls_item> type crmt_aui_sorting.
data: ls_param type zcsc_inbox_parameters.
data: lv_count type i.
data: lr_prop type ref to cl_crm_bol_entity,
      lr_entity type ref to cl_crm_bol_entity,
      lv_value type string,
      lv_guid type crmt_object_guid.

data: ls_index type ZCSCTRANSIDX.

ls_param = me->zzmap_sort_fields( it_parameters = i_query_parameters ).

lr_entity = c_native_items_found->if_bol_entity_col~get_first( ).
lv_count = 0.

while lr_entity is bound.
  lv_count = c_native_items_found->if_bol_bo_col~get_current_index( ).
  read table c_inbox_items assigning <ls_item> index lv_count.
  if sy-subrc = 0.
    try.
      lr_prop = lr_entity->get_related_entity( iv_relation_name = 'BTOrderHeader' ).
      catch: cx_crm_genil_model_error.
    endtry.
    if lr_prop is bound.
      lv_value = lr_prop->get_property_as_string( 'GUID' ). lv_guid =
lv_value.
      select single * into ls_index
        from ZCSCTRANSIDX
        where guid = lv_guid.
      if sy-subrc = 0.
        <ls_item>-MAINCATEGORY = ls_index-MAINCATEGORY.
        <ls_item>-ASSIGNEDTO = ls_index-ASSIGNEDTO.
        <ls_item>-PRIORITY = ls_index-PRIORITY.
        <ls_item>-STATUS = ls_index-STATUS.
        <ls_item>-DESCRIPTION = ls_index-DESCRIPTION.
        <ls_item>-POSTING_DATE = ls_index-CREATED_AT.
        <ls_item>-DUE_DATE = ls_index-DUE_DATE.
        <ls_item>-ZZPLANT = ls_index-ZZPLANT.
        <ls_item>-ZZCATEGORYSTR = ls_index-ZZCATEGORY1.
        <ls_item>-ZZCUSTOMER = ls_index-ACCOUNT.
        select single mc_name1 into <ls_item>-ZZNAME1
          from but000
          where partner = ls_index-account.

        endif.
      endif.
    endif.
    lr_entity = c_native_items_found->if_bol_entity_col~get_next( ).
  endwhile.
endmethod.
```

### PLEASE NOTE:

This does not mean your Custom Fields will appear in the result list. Additional BOL/UI work, not described here, has to be done as well.

## 5.4 Custom\_hit\_list\_sort

I added a Combo Box on the UI to allow a selection on ascending and descending sort. This method implements this sort logic.

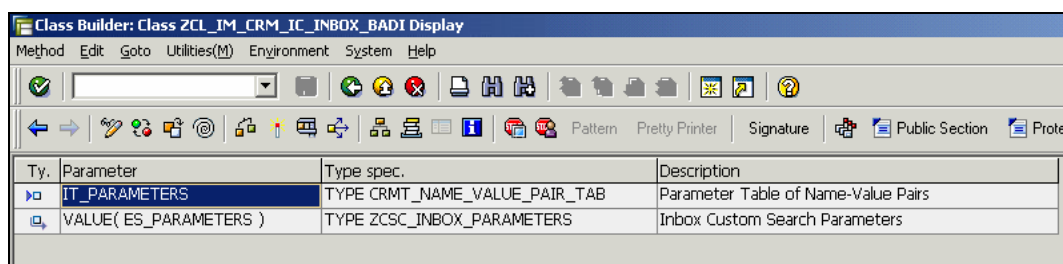
```
method if_ex_crm_ic_inbox_badi~custom_hit_list_sort.
  data: ls_param type zcsc_inbox_parameters.

  ls_param =
    me->zzmap_sort_fields( it_parameters = i_query_parameters ).

  if not ls_param-sorted_by1 is initial.
    if not ls_param-sorted_by2 is initial.
      if ls_param-zsort = 'D'.
        sort c_inbox_items by (ls_param-sorted_by1) descending
                           (ls_param-sorted_by2) descending.
      else.
        sort c_inbox_items by (ls_param-sorted_by1)
                           (ls_param-sorted_by2).
      endif.
    else.
      if ls_param-zsort = 'D'.
        sort c_inbox_items by (ls_param-sorted_by1) descending.
      else.
        sort c_inbox_items by (ls_param-sorted_by1).
      endif.
    endif.
  endif.
endmethod.
```

## 5.5 Zzmap\_sort\_fields

This is an internal method defined in the Badi to make parameter processing easier, especially for sorting.



```
method zzmap_sort_fields.
  field-symbols <ls_param> type crmt_name_value_pair.

  clear es_parameters.
  loop at it_parameters assigning <ls_param>.
    case <ls_param>-name.
      when 'ZZPLANT'.
        es_parameters-zzplant = <ls_param>-value.
      when 'ZZCATEGORY1'.
        es_parameters-zzcategory1 = <ls_param>-value.
      when 'ZZCATEGORY2'.
        es_parameters-zzcategory2 = <ls_param>-value.
    endcase.
  endloop.
endmethod.
```

```

when 'ZZSALESORG'.
  es_parameters-zzsalesorg = <ls_param>-value.
when 'SORTEDBY'.
  case <ls_param>-value.
    when '0001'. " Category
      es_parameters-sorted_by1 = 'MAINCATEGORY'.
    when '0002'. " Assigned To
      es_parameters-sorted_by1 = 'ASSIGNEDTO'.
    when '0003'. " Priority
      es_parameters-sorted_by1 = 'PRIORITY'.
    when '0004'. " Status
      es_parameters-sorted_by1 = 'STATUS'.
    when '0005'. " Description
      es_parameters-sorted_by1 = 'DESCRIPTION'.
    when '0006'. " Creation Date
      es_parameters-sorted_by1 = 'POSTING_DATE'.
    when '0007'. " Due Date
      es_parameters-sorted_by1 = 'DUE_DATE'.
    when '9001'.
      es_parameters-sorted_by1 = 'ZZPLANT'.
    when '9002'.
      es_parameters-sorted_by1 = 'ZZCATEGORYSTR'.
    when '9003'.
      es_parameters-sorted_by1 = 'ZZCUSTOMER'.
    when '9004'.
      es_parameters-sorted_by1 = 'ZZNAME1'.
  endcase.
when 'SORTEDBY2'.
  case <ls_param>-value.
    when '0001'. " Category
      es_parameters-sorted_by2 = 'MAINCATEGORY'.
    when '0002'. " Assigned To
      es_parameters-sorted_by2 = 'ASSIGNEDTO'.
    when '0003'. " Priority
      es_parameters-sorted_by2 = 'PRIORITY'.
    when '0004'. " Status
      es_parameters-sorted_by2 = 'STATUS'.
    when '0005'. " Description
      es_parameters-sorted_by2 = 'DESCRIPTION'.
    when '0006'. " Creation Date
      es_parameters-sorted_by2 = 'POSTING_DATE'.
    when '0007'. " Due Date
      es_parameters-sorted_by2 = 'DUE_DATE'.
    when '9001'.
      es_parameters-sorted_by2 = 'ZZPLANT'.
    when '9002'.
      es_parameters-sorted_by2 = 'ZZCATEGORYSTR'.
    when '9003'.
      es_parameters-sorted_by2 = 'ZZCUSTOMER'.
    when '9004'.
      es_parameters-sorted_by2 = 'ZZNAME1'.
  endcase.
when 'ZSORT'.
  es_parameters-zsort = <ls_param>-value.
endcase.
endloop.

if es_parameters-sorted_by1 is initial
  and not es_parameters-sorted_by2 is initial.
  es_parameters-sorted_by1 = es_parameters-sorted_by2.
  clear es_parameters-sorted_by2.

```

```
endif.

if es_parameters-zsort is initial.
    es_parameters-zsort = 'D'.
endif.
```

endmethod.

## 5.6 ZCSC\_INBOX\_PARAMETERS

The definition of the structure used above.

Component	RT...	Component type	Data Type	Length	Decim...	Short Description
<u>ZZPLANT</u>	<input type="checkbox"/>		STRING	0	0	
<u>ZZCATEGORY1</u>	<input type="checkbox"/>		STRING	0	0	
<u>ZZCATEGORY2</u>	<input type="checkbox"/>		STRING	0	0	
<u>ZZSALESORG</u>	<input type="checkbox"/>		STRING	0	0	
<u>SORTED_BY1</u>	<input type="checkbox"/>		STRING	0	0	
<u>SORTED_BY2</u>	<input type="checkbox"/>		STRING	0	0	
<u>ZSORT</u>	<input type="checkbox"/>		STRING	0	0	