Cookbook SAP REA Data Model extensibility

Applies to:

SAP Recycling Administration (SAP REA) on EhP6 or higher

Summary

This document explains how to extend the existing REA data model so that the extensions can be used in dialogs, checks and the declaration system.

Author: Bernd Roedel

Company: SAP AG

Created on: 1 July 2012

Author Bio

Dr. Bernd Roedel joined SAP SI in 2000. Later he moved to SAP AG and became a Development Architect. His responsibilities include the technical governance of SAP Recycling Administration. He has also worked on the SAP Enterprise Portal and in Java and Objective C projects.

Table of Contents

Preliminary Remark 3

Changes 3

Introduction to the REA master data model 4

Extending the REA Data Model 6

Extension Points 6

User Exits 8

REA Article Dialog Exit 8

REA Article Consistency Check Exit 9

REA Billing Document Data Collector Exit 9

Demo Scenario 9

Business Case 9

Defining an Data Model Extension to the Article Header 10

New DDIC Data Element / Domain 10

New Append to Structure J_7L_ARTICLE_HEADER_APP 10

Implementing the Article Dialog User Exit 10

Design and Data Flow 10

Preparation 11

New Function Module Encapsulating the User Exit’s Subscreen 11

New Class Implementing Interface J_7L_IART_DIALOG_EXIT 12

Activating the Article Dialog User Exit 13

Implementing the Consistency Check User Exit 14

New Function Module as copy of J_7LCHECK_ART_TEMPLATE 14

Activating Consistency Check User Exit 14

Implementing the Billing Document Data Collector User Exit 15

New Function Module as copy of J_7L_VBRP_FILTER_EXIT 15

Activating the Billing Document Data Collector User Exit 16

Testing the Scenario 17

Appendix .... **Error! Bookmark not defined.**

Related Content 18

Copyright 19

Preliminary Remark

The following document is a cookbook for extending the REA data model in customer namespace. It provides an overview on the “extension points” and explains in example how data model extension and REA user exits help to enhance an REA instance according to individual customer requirements. This cookbook does not explain in depth how to use the ABAP workbench or how to create DDIC structures.

Changes

This is the initial version.

Introduction to the REA master data model

The REA master data consists of two main objects: REA **article** and REA (**packaging) component**.

A REA (packaging) component is based on REA **internal fractions**. Figure 1 depicts the master data model schematically. The colors in Figure 1 will be used consistently throughout this document.

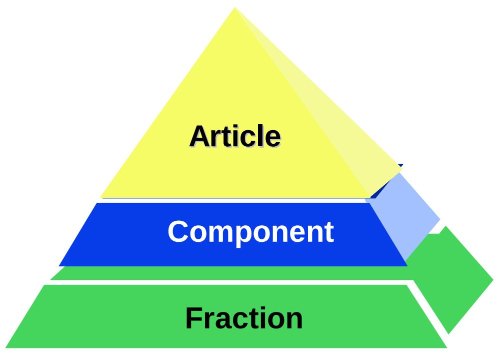


Figure 1: REA Master data scheme

REA article and REA components refer to the material number in the ERP Material master (MARA-MATNR). Hence REA article cannot exist without a corresponding MM entry. REA packaging components may exist without a material master entry, if configured appropriately. In addition it also is possible that a REA article and a REA component refer to the identical MM entry (Figure 2).

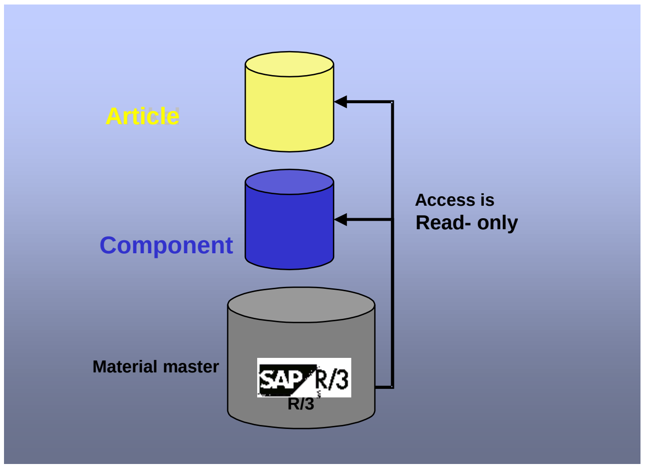


Figure 2: Relationship to ERP Material master

REA internal fractions are maintained in the REA customizing and assigned to one or many recycling partner fractions (Figure 3). It is also possible to assign an internal fraction to a recycling partner fraction without specifying a partner fraction. In this case there is no settlement with this particular recycling partner for that internal fraction. This mechanism is called **cancellation of partner fraction requirement**. This fraction assignment is out of scope for this document.

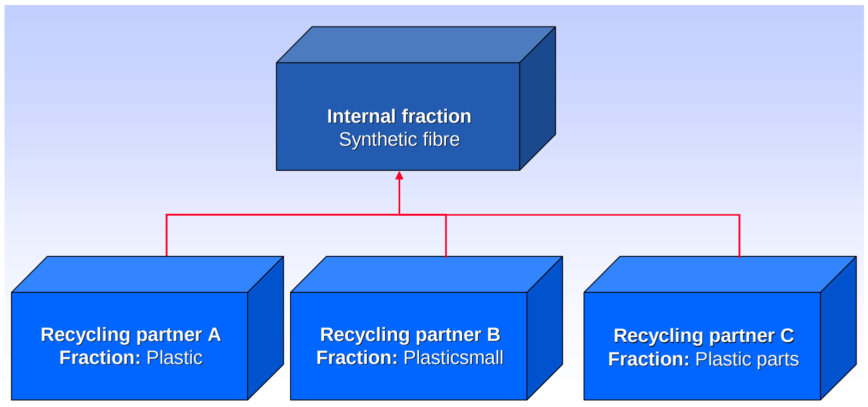


Figure 3: Assignment internal fraction to recycling partner fraction by Customizing

An internal fraction in REA is a packaging material that can be assigned 1 to n times to a REA packaging with a defined weight / weight unit.

A REA component consists of one or many internal fractions and that can be settled with one or many recycling partners. Both assignments are time dependent, so that several packaging versions with a non-overlapping timeframes can be created. A REA component can be of type **consumed packaging**, which can be directly identified in material movements by the declaration system, or of type **sales packaging**, which can be assigned to one or many REA articles as a component.

A REA article represents finished product that must be reported to a recycling partner due to the legal obligations of the REA user. In addition to the material number, a REA article is identified by the key values company code, country and sales unit. REA components and recycling partners are assigned to REA articles in a time dependent manner. A REA article is directly identified in billing documents and/or material movements by the declaration system.

Figure 4 depicts the REA master data structuring. The REA article and the REA components are the two main components that form the REA master data. In addition to the keys and relationships explained so far, REA articles and REA components can hold various attribute values that are generally dependent on the assigned recycling partner.

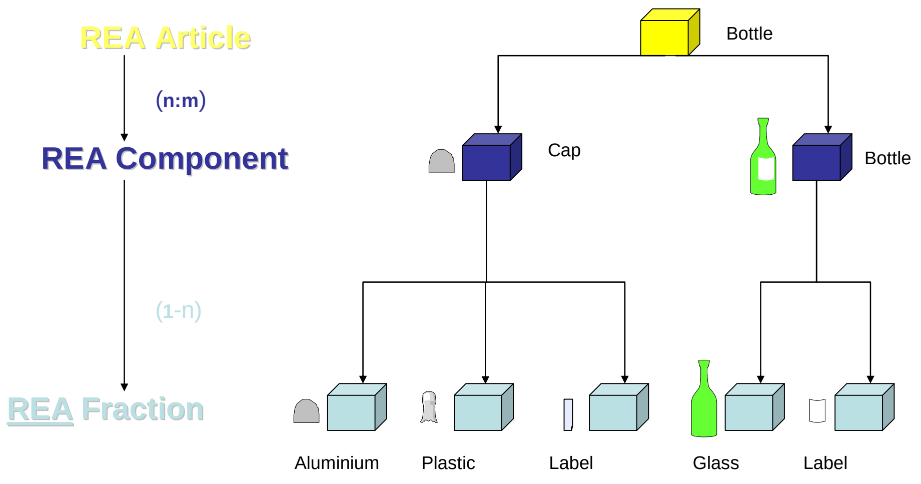


Figure 4: REA Master data

REA components are maintained by the transactions J7L5/J7L6/J7L7. The tab **fraction** is used to maintain the internal fraction assignment. The tab **partner** is used to maintain the recycling partner assignment. A REA component can only be settled with partners that are assigned to the packaging in a particular timeframe.

REA articles are maintained by transactions J7L1/J7L2/J7L3. The tab **packaging** is used to maintain the REA component assignment. The tab partner is used to maintain the recycling partner assignment. A REA article can only be settled with partners that are assigned to the article in a particular timeframe. In case there are multiple recycling partners assigned in the same timeframe, **license fee splitting** rules enable the user to split the license fee of the packaging between the assigned recycling partners dependent on the business process.

REA articles can alternatively be maintained as **reference articles** by transaction J7L0. Reference articles are settled exactly as their referenced article. Reference articles are not discussed in detail in this document.

Figure 5 summarizes the REA master data maintenance process.

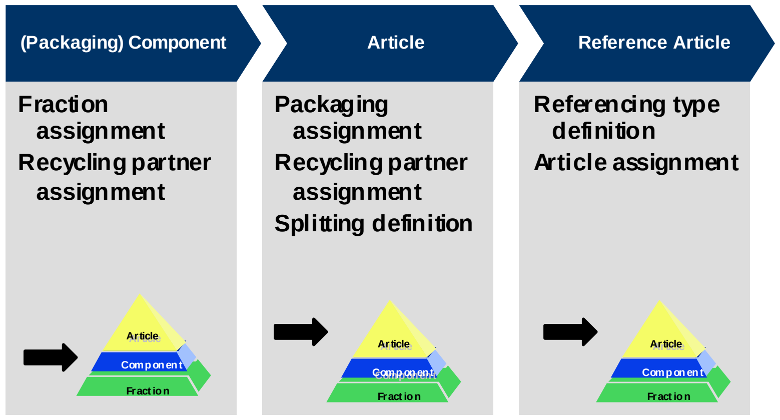


Figure 5: Process view

Extending the REA Data Model

Extension Points

REA provides a list of extension points that provide an „anchor" to extend the REA data model. Using the extension points has got the advantage that the user does not have to care about each and every REA DB table, structure, program etc. that has to be extended to implement a certain business scenario. If the extension point is used, the extension is automatically included in the necessary structures and programs. Figure 6 shows the extension points within the REA data model introduced in section above. All extensions are numbered and their position in the data model is drawn as lollypop. Yellow lollypops illustrate the REA article extension points; blue lollypops illustrate the REA packaging extension points.

Using extension point number 1 means extending REA article information on the article header level. Using extension point number 8 means extending REA packaging information on the level of the recycling partner assignment to packaging.

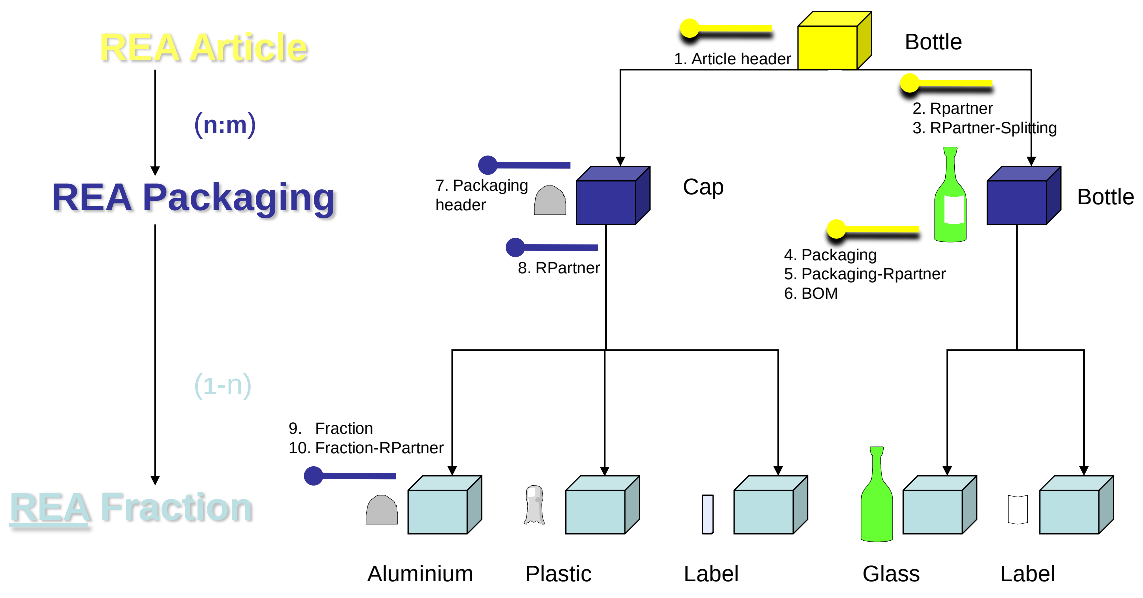


Figure 6: Extension Point in REA Master Data Model indicated by a Numbered and Colored Lollypop Symbol

Table 1 links the extension points to the DDIC structures. Each structure has got a suffix to avoid naming conflicts. Renaming can be used to address the structures' content easily. The DDIC structures are the technical representation of the anchor. For technical reasons they contain one field as placeholder. They can be used to extend the REA data model to the desired level, by appending them with fields in customer namespace.

Lollypop Number	Structure in DDIC	Suffix	Renaming	Function
1	J_7L_ARTICLE_HEADER_APP	hea	header	Article header data
2	J_7L_ARTICLE_PARTNER_APP	par	partner	Partner assignment to Article
3	J_7L_ARTICLE_PARTNERSPLIT_APP	pas	partnersplit	Partner splitting assignment to article
4	J_7L_ARTICLE_PACK_APP	pac	pack	Packaging assignment to article
5	J_7L_ARTICLE_PACKPARTNER_APP	ppa	pacpartner	Partner dependent Packaging assignment to article
6	J_7L_ARTICLE_BOM_APP	bom	bom	BOM assignment to article
7	J_7L_PACK_HEADER_APP	hea	header	packaging header data
8	J_7L_PACK_PARTNER_APP	par	partner	Partner assignment to packaging
9	J_7L_PACK_fraction_APP	fra	fraction	Fraction assignment to packaging
10	J_7L_pack_fracPARTNER_APP	frp	fractionpartner	Partner dependent Fraction assignment to packaging

Table 1: Extension Point List

In case I would like to use extension point 1 to extend the REA article information on the article header level, I have to create an append to DDIC structure J_7L_ARTICLE_HEADER_APP.

When planning a data model extension the second challenge is to leverage the new information provided by the extended data model in the business process. Therefore the next section will introduce various user exits provided by REA.

User Exits

In general SAP note 891715 gives on overview on the user existing in REA. In our example we will focus on the REA article dialog exit, on the exit of the article consistency check and on the exit of billing document data collector in the declaration system.

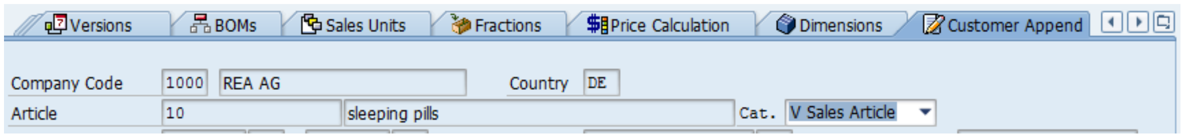
REA Article Dialog Exit

Prerequisite is the usage of controls in REA master data dialogs.

This exit produces, when implemented, an additional tab in the REA article dialog (Figure 7). The user exit can display its own content in a dynpro subscreen on this tab. Technically the user exit is defined by the interface J_7L_IART_DIALOG_EXIT. This interface offers full read/write access to the master data of the REA article by reference semantics. In addition the user exit interface may communicate with the REA article dialog by several methods of the interface:

Method	
IS_CONSISTENT	Is called prior to saving the article. The method may prevent the article from being saved by signaling an inconsistent state.
IS_CHANGED	Is called prior to leaving the article dialog. The method may prevent the loss of data by signaling that saving the article is required.
SAVE	Is called when the article is saved and can be used to persist data outside the REA database tables.
REQUESTS_FOCUS	Is called when the user changes tabs. The method may used to direct the focus to the tab implemented by the user exit.
Event	
CONTENT_UPDATE	Signals that the internal data is updated by the user exit implementation and that the controls showing packaging and partner assignments need to be updated.

Table 2: Methods/Events of J_7L_IART_DIALOG_EXIT



Company Code	1000	REA AG	Country	DE
Article	10	sleeping pills	Cat.	V Sales Article

Figure 7: REA Article Dialog with Implemented User Exit on Right Hand Side

REA Article Consistency Check Exit

This user exit is assigned to a particular recycling partner. It is executed within the article consistency check. This happens when the article is saved, when the assigned recycling partners are displayed, when the function consistency check is called by the user or when a collective article consistency check is executed for the recycling partner. The user exit is defined as function module with an interface identical to function module J_7LCHECK_ART_TEMPLATE. The function module's import parameter provides full read access to the article's data in question. The changing parameters signal the check's result plus an error text, in case the check fails. The text appears as part of a message, when the check's results are displayed.

REA Billing Document Data Collector Exit

This user exit is defined in general for REA billing document data collector in the declaration system. It may also take into account the recycling partner, company code, country, document type of the declaration, if required. The user exit is executed each time the data collector reads one billing document item. It is implemented as function module with an interface identical to function module J_7L_VBRP_FILTER_EXIT. The function module's import parameters provide full read access to the billing document header and item in question. The changing parameters signal the evaluation result plus a sign, in case the billing document item is scheduled to further processing.

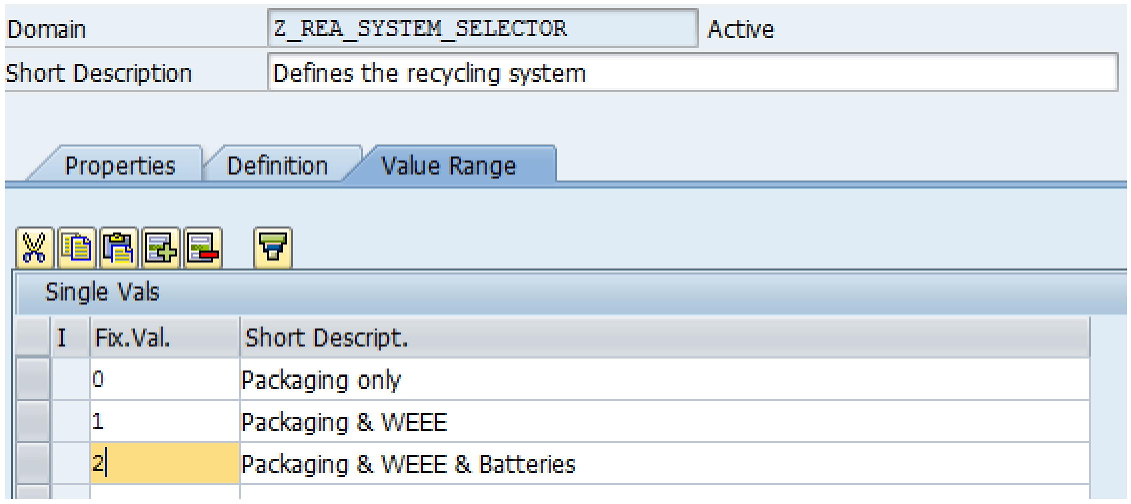
Demo Scenario

Business Case

The demo scenario's business case entails the idea to mark REA articles, whether they are relevant for packaging only settlement, for packaging plus WEEE settlement or for packaging plus battery plus WEEE settlement. In REA this is called the recycling system. Marking shall be executed on the article header level (material number, company code, country, sales unit). The mark shall be used in the declaration system and consistency check to ensure a high data quality plus a full compliance to the legal requirements. The consistency check validates the mark with the material group in MM, because in our business case the material group determines the recycling system. The declaration system filters out the item, in case the mark does not match the recycling partner of the declaration. This improves performance and ensures we have only article in the declaration's log that match the recycling system of the declaration's recycling partner.

Defining an Data Model Extension to the Article Header

New DDIC Data Element / Domain

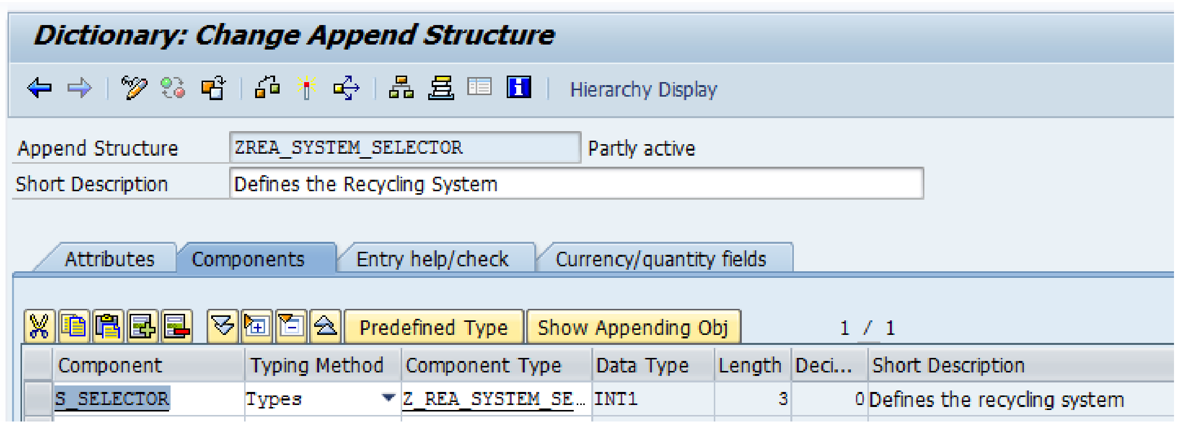


Domain	Z_REA_SYSTEM_SELECTOR	Active
Short Description	Defines the recycling system	

I	Fix.Val.	Short Descript.
	0	Packaging only
	1	Packaging & WEEE
	2	Packaging & WEEE & Batteries

Figure 8: Creating a New Domain Z_REA_SYSTEM_SELECTOR in DDIC with Type INT1 with fixed Values (Packaging only is default)

New Append to Structure J_7L_ARTICLE_HEADER_APP



Dictionary: Change Append Structure

Append Structure	ZREA_SYSTEM_SELECTOR	Partly active
Short Description	Defines the Recycling System	

Component	Typing Method	Component Type	Data Type	Length	Deci...	Short Description
S_SELECTOR	Types	Z_REA_SYSTEM_SE...	INT1	3	0	Defines the recycling system

Figure 9: Creating an Append with Component s_selector

It is useful to set the enhancement category of the append to *cannot be enhanced* to avoid problems during activation.

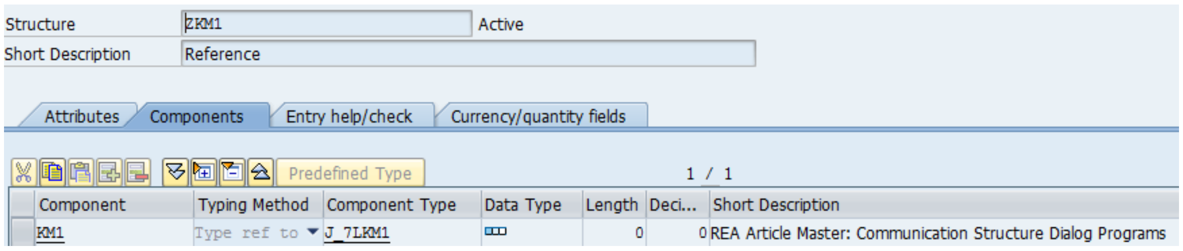
Implementing the Article Dialog User Exit

Design and Data Flow

This example is based on the concept of update by reference semantics. The user exit implementing class is called with a reference to J_7LKM1 (IO_KM1) containing all article header data and of course our append from section above. The reference is stored as private attribute KM1 and passed on as changing parameter IO_KM1 to a function module. The function module stores the reference globally in the top include G_KM1 of the function group. During every PBO of the subscreen the value of our append is moved from the reference to a local variable WA_KM1. During every PAI the value is moved back to the global reference G_KM1 (eventually back to the article dialog).

Preparation

First we create a new structure in DDIC ZKM1 that encapsulates a reference to structure J_7LKM1.



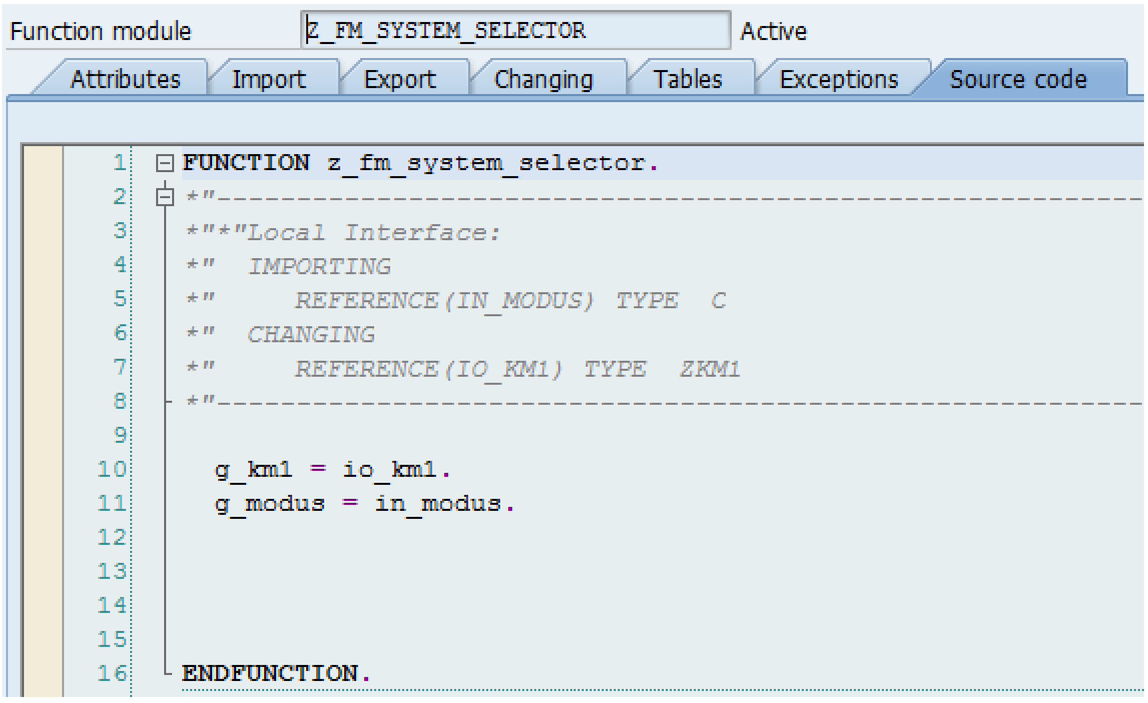
Structure: ZKM1 Active
Short Description: Reference

Component	Typing Method	Component Type	Data Type	Length	Deci...	Short Description
KM1	Type ref to	J_7LKM1		0	0	REA Article Master: Communication Structure Dialog Programs

Figure 10: DDIC Structure Storing a Reference to J_7LKM1

New Function Module Encapsulating the User Exit's Subscreen

In the next step we will create a function module Z_FM_SYSTEM_SELECTOR in a function group **<zmy_group>** that encapsulates the subscreen displayed by the user exit.



Function module: Z_FM_SYSTEM_SELECTOR Active

```
FUNCTION z_fm_system_selector.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(IN_MODUS) TYPE  C
*"  CHANGING
*"     REFERENCE(IO_KM1) TYPE  ZKM1
*"----------------------------------------------------------------------


  g_km1 = io_km1.
  g_modus = in_modus.





ENDFUNCTION.
```

Figure 11: Function Module serving as Interface to Function Group Global Variables

Function group **<zmy_group>** has got following variables defined in the TOP include:

```
DATA: g_km1 TYPE zkm1, "global reference to J_7LKM1
      wa_km1 TYPE j_7lkm1, "local representation of J_7LKM1
      g_modus TYPE c. " X=> change mode/space  = display mode
```

Next we create dynpro 1000 in function group **<zmy_group>** containing the following flow logic :

```
PROCESS BEFORE OUTPUT.
  MODULE init_1000.
*
PROCESS AFTER INPUT.
  MODULE USER_COMMAND_1000.
```

The PBO ensures that the global data is passed to the local data and takes care of change/display mode. Screen-group1 is used to identify input fields:

```
*&---------------------------------------------------------------------*
*&      Module  INIT_1000  OUTPUT
*&---------------------------------------------------------------------*
*  Transfers global values to Dynpro
*----------------------------------------------------------------------*
MODULE init_1000 OUTPUT.
  MOVE g_km1-km1->* TO wa_km1.

  LOOP AT SCREEN.
    IF screen-group1 = '001' AND g_modus = space.
      screen-input = 0.
      MODIFY SCREEN.
    ENDIF.
  ENDLOOP.
ENDMODULE.                 " INIT_1000  OUTPUT
```

The PAI ensures that the local data is passed back to the global data.

```
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_1000  INPUT
*&---------------------------------------------------------------------*
*   Transfers dynpro values to global values
*----------------------------------------------------------------------*
MODULE user_command_1000 INPUT.
  MOVE-CORRESPONDING wa_km1 TO g_km1-km1->*.
ENDMODULE.                 " USER_COMMAND_1000  INPUT
```

The dynpro itself contains a text field and in input/output field defined as dropdown **`Listbox`**. Screen group 1 is set to **`001`**. The dropdown listbox refers to the local variable WA_KM1-S_SELECTORHEA. (<variable> -<componentname> <suffix>).

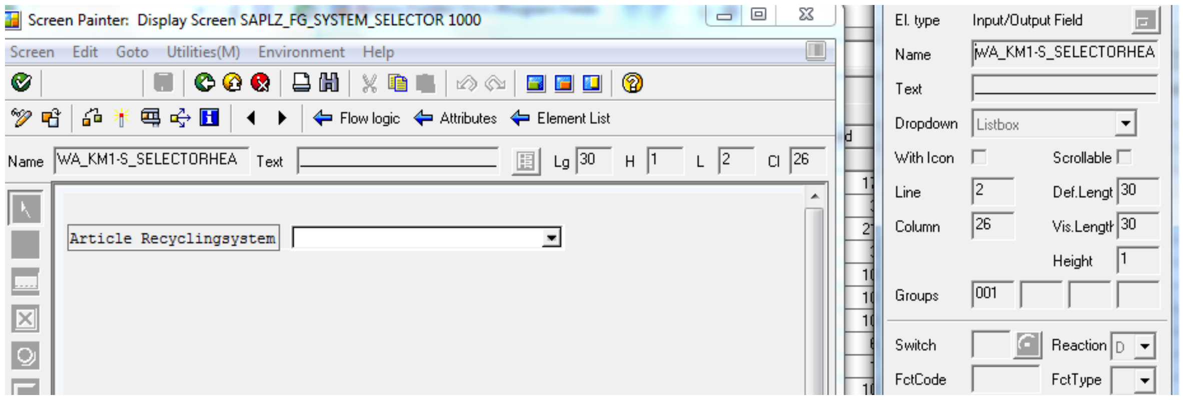
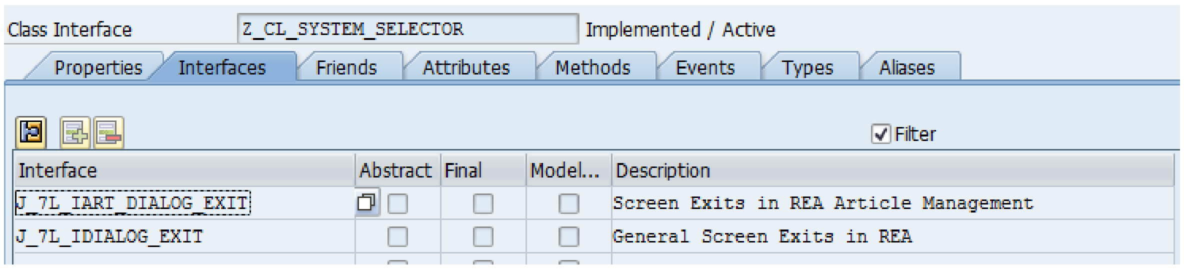

Screen Painter: Display Screen SAPLZ_FG_SYSTEM_SELECTOR 1000
Screen Edit Goto Utilities(M) Environment Help
Flow logic
Attributes
Element List
Name WA_KM1-S_SELECTORHEA Text Lg 30 H 1 L 2 Cl 26
Article Recyclingsystem
El. type Input/Output Field
Name WA_KM1-S_SELECTORHEA
Text
Dropdown Listbox
With Icon
Scrollable
Line 2
Def.Lengt 30
Column 26
Vis.Length 30
Height 1
Groups 001
Switch
Reaction D
FctCode
FctType

Figure 12: Subscreen Dynpro Definition

New Class Implementing Interface J_7L_IART_DIALOG_EXIT

First we create a new class Z_CL_SYSTEM_SELECTOR that implements the interface J_7L_IART_DIALOG_EXIT.



Class Interface Z_CL_SYSTEM_SELECTOR Implemented / Active

Interface	Abstract	Final	Model...	Description
J_7L_IART_DIALOG_EXIT	☐	☐	☐	Screen Exits in REA Article Management
J_7L_IDIALOG_EXIT	☐	☐	☐	General Screen Exits in REA

1. A private instance variable **km1** of type **zkm1** must be defined first.
2. Method INITIALIZE must be fully implemented:

```
me->km1-km1 = io_km1.
  me->j_7l_iart_dialog_exit~dynprogram = 'SAPL<zmy_group>'.
  me->j_7l_iart_dialog_exit~dynnr = '1000'.
  me->j_7l_iart_dialog_exit~tabtext = 'Recyc. system'.
  me->j_7l_iart_dialog_exit~tabicon = icon_change_text.
  me->j_7l_iart_dialog_exit~tabtooltext = 'Recyc. system definition'.

CALL FUNCTION 'Z_FM_SYSTEM_SELECTOR'
  EXPORTING
    in_modus        = in_mode
  changing
    io_km1          = me->km1.
```

The method defines the text/icons on the screen and dynpro number in function group **<zmy_group>** of section “New Function Module Encapsulating the User Exit’s Subscreen” that represents the subscreen.

The method IS_CONSISTENT should return ABAP_TRUE. The method IS_CHANGED should return ABAP_FALSE, because by using the extension point we also use the check changes mechanism of the article dialog. The method REQUESTS_FOCUS should return ABAP_FALSE. The method SAVE does not have to be implemented, because by using the extension point we also use the REA article API for persistency.

Activating the Article Dialog User Exit

You can activate the use exit in REA customizing → adaption -> define customer specific enhancements by entering a line with key SAPMJ7LM 8 and the class name from section “New Class Implementing Interface J_7L_IART_DIALOG_EXIT”.

Implementing the Consistency Check User Exit

New Function Module as copy of J_7LCHECK_ART_TEMPLATE

Create a copy of FM J_7LCHECK_ART_TEMPLATE as FM Z_CHECK_SYSTEM_SELECTOR in function group <zmy_group>. The following coding validates the recycling system against the material group in the material master:

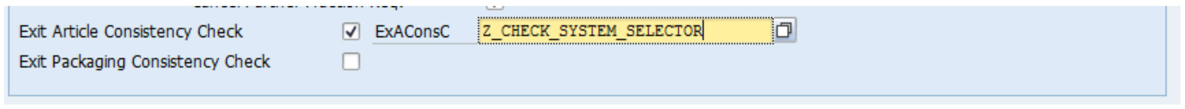
```
FUNCTION z_check_system_selector.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(I_ART) TYPE  J7LR1_ARTIKEL_LINE
*"  TABLES
*"      I_PACK TYPE  J7LR1_VERPACKUNG_TAB
*"  CHANGING
*"     VALUE(IO_OK) TYPE  C
*"     REFERENCE(IO_ERROR_TEXT) TYPE  C DEFAULT SPACE
*"----------------------------------------------------------------------

  DATA l_mara TYPE mara.
  SELECT SINGLE * INTO l_mara
    FROM mara
    WHERE matnr = i_art-matnr.
  io_ok = 'X'.
  CASE i_art-s_selectorhea.
    WHEN 0.
      IF l_mara-matkl <> '01'.
        CLEAR io_ok.
        io_error_text = 'Recyc. System is packaging only'.
      ENDIF.
    WHEN 1.
      IF l_mara-matkl <> '02'.
        CLEAR io_ok.
        io_error_text = 'Recyc. System is packaging& WEEE'.
      ENDIF.
    WHEN 2.
      IF l_mara-matkl <> '03'.
        CLEAR io_ok.
        io_error_text = 'Recyc. System is packaging& WEEE Battery'.
      ENDIF.
  ENDCASE.

ENDFUNCTION.
```

The coding compares the entered recycling system in REA (field S_SELECTORHEA) to the material group of the article. The comparison uses constants defined in the coding.

Activating Consistency Check User Exit

You can activate the use exit in REA customizing → adaption → define customer specific enhancements by entering a line with key **SAPMJ7LE 5** and the function group name **<zmy_group>** from section “New Function Module as copy of J_7LCHECK_ART_TEMPLATE”.

Now you can activate the user exit for each recycling partner separately in transaction J7LE by activating the indicator and entering the FM defined in section 0 (Z_CHECK_SYSTEM_SELECTOR).



Exit Article Consistency Check	✓	ExAConsC	Z_CHECK_SYSTEM_SELECTOR
Exit Packaging Consistency Check	☐		

The exit is only executed for those recycling partner.

Implementing the Billing Document Data Collector User Exit

New Function Module as copy of J_7L_VBRP_FILTER_EXIT

Create a copy of FM J_7L_VBRP_FILTER_EXIT as FM Z_VRP_SYSTEM_SELECTOR in function group **`<zmy_group>`**. Replace the existing coding with the following lines of code:

```
  DATA: e09 TYPE j_7le09,
        m03 TYPE j_7lm03,
        xfield TYPE c. "dummy
* 1. Read article header
  SELECT SINGLE * INTO m03
    FROM j_7lm03
    WHERE matnr = vbrp_in-matnr
    AND  bukrs = bukrs
    AND  land1 = land1
    AND  vrkme = vbrp_in-vrkme
    AND lvormadm = space.

* 2. Get context from memory
  IMPORT j_7le09_save = e09 FROM MEMORY  ID 'J_7LVFV1'.
* 3. Evaluate system_selector field in header
  CASE m03-s_selectorhea.
    WHEN 0.
      SELECT SINGLE b~kz_packaging INTO xfield
        FROM j_7le01 AS a
        JOIN j_7lc25 AS b
        ON a~fmgrp = b~fmgrp
        WHERE a~entnr = e09-entnr
        AND   a~land1 = e09-land1
        AND   a~lvormadm = space
        AND b~kz_packaging = 'X'.
      IF sy-subrc <> 0. "pack only
        CLEAR filter_ok.
      ENDIF.
    WHEN 1.
      SELECT SINGLE b~kz_packaging  INTO xfield
    FROM j_7le01 AS a
    JOIN j_7lc25 AS b
    ON a~fmgrp = b~fmgrp
    WHERE a~entnr = e09-entnr
    AND   a~land1 = e09-land1
    AND   a~lvormadm = space
    AND ( b~kz_packaging = 'X'
        OR b~kz_weee = 'X' ).
      IF sy-subrc <> 0. "pack or WEEE
        CLEAR filter_ok.
      ENDIF.
    WHEN 2.
      SELECT SINGLE b~kz_packaging  INTO xfield
   FROM j_7le01 AS a
   JOIN j_7lc25 AS b
   ON a~fmgrp = b~fmgrp
   WHERE a~entnr = e09-entnr
   AND   a~land1 = e09-land1
   AND   a~lvormadm = space
   AND ( b~kz_packaging = 'X'
       OR b~kz_weee = 'X'
       OR b~kz_battery = 'X' ).
      IF sy-subrc <> 0. "pack or WEEE or battery
        CLEAR filter_ok.
      ENDIF.
  ENDCASE.
```

The coding reads the recycling system (field in extension point) and compares it to the recycling system of the declaration's recycling partner, which is derived from the FORMGROUP settings in REA customizing.

Activating the Billing Document Data Collector User Exit

Go to REA customizing → general control → define general control and enter the FM Z_VRP_SYSTEM_SELECTOR of section into field exit function billing Items (Figure 13).

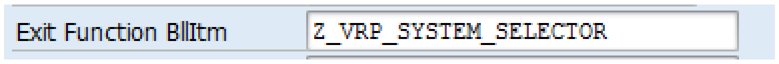
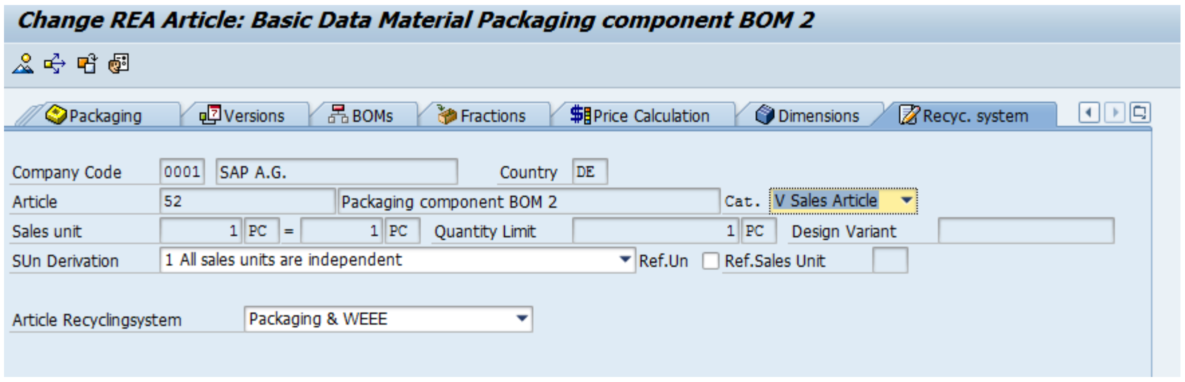

Exit Function BllItm Z_VRP_SYSTEM_SELECTOR

Figure 13: User Exit in Define General Control

Testing the Scenario

Figure 14 shows the new tab when entering the REA article dialog in change mode. The dropdown box allows entering the recycling system of the article.



Change REA Article: Basic Data Material Packaging component BOM 2

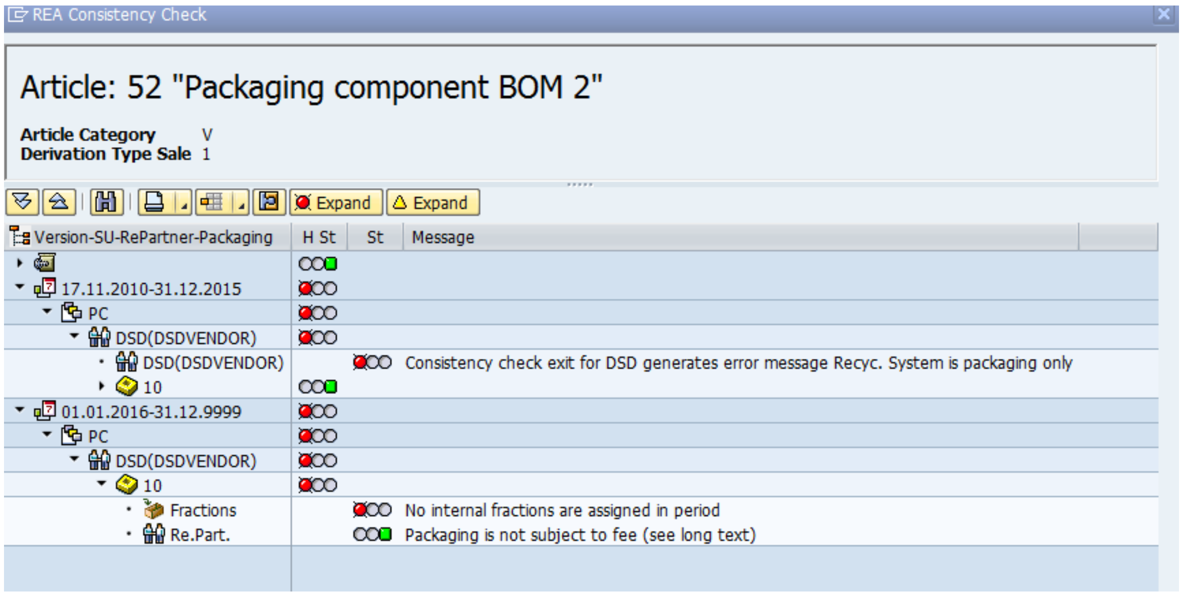
Tabs: Packaging | Versions | BOMs | Fractions | Price Calculation | Dimensions | Recyc. system

Company Code	0001 SAP A.G.	Country	DE
Article	52 Packaging component BOM 2	Cat.	V Sales Article
Sales unit	1 PC = 1 PC	Quantity Limit	1 PC
SUn Derivation	1 All sales units are independent	Ref.Un	Ref.Sales Unit
Article Recyclingsystem	Packaging & WEEE		

Design Variant

Figure 14: New Tab within the Article Dialog

Figure 15 shows the consistency check for recycling partner DSD after activating the user exit for DSD. The user exit message is displayed, in case the check fails.



REA Consistency Check

Article: 52 "Packaging component BOM 2"

Article Category V
Derivation Type Sale 1

Version-SU-RePartner-Packaging	H St	St	Message
	○○🟩		
17.11.2010-31.12.2015	🟥○○		
PC	🟥○○		
DSD(DSDVENDOR)	🟥○○		
DSD(DSDVENDOR)		🟥○○	Consistency check exit for DSD generates error message Recyc. System is packaging only
10	○○🟩		
01.01.2016-31.12.9999	🟥○○		
PC	🟥○○		
DSD(DSDVENDOR)	🟥○○		
10	🟥○○		
Fractions		🟥○○	No internal fractions are assigned in period
Re.Part.		○○🟩	Packaging is not subject to fee (see long text)

Figure 15: New Message in Consistency Check

If the declaration is started for billing documents, the user exit for the declaration is executed by the declaration system. The billing document item rejection by the user exit is visible in the single log protocol only, if the settings for protocol are set to *log filter item* (REA customizing → general control → define general control or document type detail screen).

Related Content

[1] Cookbook_Rpartner

Copyright