

# BRFplus for Demand Side Management



## Applies to:

Business experts and IT professionals that are considering automating decisions, checking data, defaulting or calculating values in business applications. A pre-requisite to apply the approaches explained in this document is SAP NetWeaver (ABAP) 7.0 Enhancement Package 2 or above. For more information, visit the [Business Rules Management homepage](#).

## Summary

The document describes, with the help of an example, how a decision service can be implemented using BRFplus. Each step from concept to final implementation is shown. The example demonstrates the automatic determination of eligibility of a Demand Side Management Program Application and/or Agreement.

**Author:** Diane Volk

**Company:** SAP Australia

**Created on:** 15 December, 2011

## Author Bio

Diane Volk is a Utilities Industry Consultant at SAP Australia. She joined SAP in 2002. Since then he has been working in various projects as a developer, functional consultant and architect.

## Table of Contents

Purpose of this Document .....	3
The Example.....	3
Steps in the implementation of a Decision Service .....	3
Definition of Task .....	4
Decision Service Definition .....	4
Decision Service Implementation.....	5
Decision Service Testing.....	13
Decision Service Trigger Implementation .....	14
The End Result .....	17
Related Content.....	18
Copyright.....	19

## Purpose of this Document

This document shows, with the help of an example, how a decision service can be implemented using BRFplus and subsequently connected to a business process.

BRFplus is a framework that allows Users to define and maintain business rules for all kinds of purposes and applications independently of the source code in SAP. BRFplus allows business logic to be defined in an intuitive way without the need to write code and facilitates reuse of business logic rules in different applications.

Example of types of business rules that BRFplus can be used to implement include:

- Data validation rules
- Business rules to determine output parameter(s) e.g. a decision table
- Calculations of various types

The primary purpose of this document is to provide an example of an actual application of BRFplus. The intention is to help business users understand how BRFplus can be utilized in their own business processes and to help IT professionals understand the steps required to implement a BRFplus decision service.

The example in this document is the evaluation of eligibility criteria of a utility customer to participate in a rewards and incentives program to reduce their energy consumption. However the approach is not limited to this particular example. Whenever a task has to be automated, BRFplus may be used to implement a decision service. The decision service can then be plugged into the relevant process step - ideally via a BADI, enhancement spot or any other exit technology. Remote invocation of BRFplus decision services is also possible by calling a web service or an RFC-enabled function module. Both can be generated from within the BRFplus workbench.

## The Example

Utility companies are increasingly focused on energy conservation and managing energy demand. Utility companies need new ways to closely monitor and influence energy demand to ensure they can provide reliable, affordable energy for their customers. As a result many companies are responding by investing more heavily in demand side management (DSM) programs, such as energy efficiency (EE) and demand response (DR) programs. Energy efficiency programs are typically one-off transactions where a customer receives incentives for changing their behaviour e.g. installing energy efficient light bulbs or purchasing and installing energy efficient appliances. Demand Response Programs target customers to change their usage patterns to achieve a more balanced load profile i.e. to avoid peaks and troughs in demand. Traditionally demand response programs were targeted at industrial and commercial customers. With the advances in metering technology (e.g. AMI), increasingly these programs are being offered on a wider basis to residential customers e.g. remote control devices to temporarily turn off air conditioners or pool pumps during periods of peak load.

The SAP solution to enable users to create and manage DSM programs is implemented as part of the SAP Utilities Industry CRM solution. The solution uses BRFplus to implement the rules that define a customer's eligibility to participate in a DSM program and to determine whether the customer has met program goals to qualify for incentives.

## Steps in the implementation of a Decision Service

There are three main steps in the implementation of a Decision Service:

- 1) Definition, i.e. defining function name and signature
- 2) Implementation, i.e. business logic
- 3) Trigger implementation, i.e. call of BRFplus function from the application

We will now look at each step in relation to the DSM example.

## Definition of Task

First clarify the task to be accomplished with the decision service. Perfect candidates are tasks that are well structured and unambiguous.

Often tasks are related to one of the following:

- Validations
- Derivations
- Defaulting
- Classification
- Calculation
- Mapping
- Boolean decisions

Typically these are micro decisions. Micro decisions occur very frequently but the impact of a particular decision is not big compared with decisions that affect the complete enterprise such as strategic decisions about starting a new product line. However, micro decisions tend to occur very frequently and cumulatively they can have a big impact on the financial result of a company as they directly impact the efficiency of the operations.

In the example of this document the task of this decision service is to determine a customer's eligibility to participate in a DSM program based on the information provided in their application. The outcome will be to set the status of the application.

## Decision Service Definition

Knowing the task we can start to define the decision service:

- Name "DSM Eligibility Check"
- Description "Checks the eligibility of a DSM Application"

We also need to define inputs (the context for the decision rules) and outputs (the results) of the service. Ideally, the equivalent Data Dictionary data elements (shown in brackets in the example below) are already known so that technical attributes as well as business semantics can be derived from the dictionary.

- Inputs/Context
  - Customer Data Structure (CRMS\_IU\_DSM\_BRF\_PARTNER\_DATA)
  - DSM Application Header (CRMS\_IU\_DSM\_BRF\_APPL\_HEADER)
  - Premise Data (CRMS\_IU\_DSM\_BRF\_PREMISE\_DATA)
- Outputs/Results
  - Eligibility Result (IS\_ELIGIBLE)

Parameters are not always elementary (i.e. single value). This is why BRFplus also supports structure parameters, table parameters, and even deeply nested complex types (e.g. a structure with a component being a table).

The service definition is shown in the next screenshot. In BRFplus a service is implemented with the help of a BRFplus Function. The Signature tab contains the context, which provides the inputs for the service call. It also contains a Result Data Object, in the example, the eligibility result. The following screenshot shows the Function in the BRFplus workbench (transaction code BRF+).

Function: ZCRM\_DSM\_AUTO\_APL\_ELIGIBILITY, Eligibility DSM Apl.

Back | Edit | Check | Save | Activate | Delete | More

General

Detail

Start Simulation

Properties | **Signature** | Assigned Rulesets

Context

Component Name	Text	Type
▶ <a href="#">CUSTOMER_DATA</a>	Contains the business partner details	Structure
▶ <a href="#">DSM_APPLICATION_HEADER</a>	Contains the general header data for DSM Application	Structure
▶ <a href="#">PREMISE_DATA</a>	Contains the premise details	Structure

Result Data Object

Data Object: [Elig.RsIt](#)

Data objects are used in BRFplus to define parameters. The parameter Customer Data, for example, is defined by a data object shown in the next screenshot. This can be reached by selecting the hyperlink on CUSTOMER\_DATA in the Signature tab. You can see that properties such as texts, element type (text), and length are taken over from the corresponding data dictionary object. BRFplus is also able to find possible values in domain lists or check tables so that a value help based on those values can be provided when modeling rules.

General

General | Texts | Documentation

Name:  Access Level:

ID:  Storage Type:

Versioning:  Application:

Created By:  Changed By:

Created On:   Changed On:

Detail

Define Data Binding

Select Bind Type:

DDIC Type Name:  Refresh Binding

Structure Definition

Component Name	Text	Type
▪ <a href="#">BUSINESS PARTNER NUMBER</a>	Long Name	Element (Text)
▪ <a href="#">BUSINESS PARTNER NAME</a>	Partner	Element (Text)
▪ <a href="#">BUSINESS PARTNER CATEGORY</a>	Part. cat.	Element (Text)

## Decision Service Implementation

The service is defined but no business logic has been defined so far.

Business logic can be implemented in Rulesets in BRFplus. One or more Rulesets can be created from the Function screen. Rulesets can have pre-conditions and priorities to define the order of execution. In the DSM Application Eligibility example only one Ruleset is defined.

**Function: CRM\_DSM\_CHECK\_APP\_ELIGIBILITY, Eligibility DSM Apl.**

Back | Edit | Check | Save | Activate | Delete | More

**General**

General | Texts | Documentation

Name: CRM\_DSM\_CHECK\_APP\_ELIGIBILITY | Access Level: Application  
 ID: 3C746F4C5166A905E1000000A4286AF | Storage Type: System | Transportable  
 Versioning: Off | Application: Example DSM App.  
 Created By: DDIC | Changed By: DDIC  
 Created On: 28.09.2010 13:45:13 | Changed On: 24.11.2010 18:39:00

**Detail**

Start Simulation

Properties | Signature | **Assigned Rulesets**

**Ruleset**

View: Assigned Rulesets | Create Ruleset

Name	Text	Priority	Enabled	Precondition
APP_ELIGIBILITY_RULESET	DSM Apl. Ruleset	[undefined]	Yes	

The Ruleset (below) is an example of a very simple rule. The rule states that if the Business Partner Category is of type 'Person' (i.e. not a company) then the result is 'Eligible' else the Business Partner is 'Ineligible' to participate in the DSM program.

**Detail**

Hide Ruleset Header | Variables | Context Overview

Enabled:  | Number of Rules: 1  
 Function: Eligibility DSM Apl. | Number of Variables: 0  
 Precondition: <Not assigned> | Priority: 00

**Rules**

(1) Rule: Only Residential Customers are Eligible - Unlimited Validity

**If**

Contains...Part\_cat is equal to 1 (Person)

**Then**

(1) Change value of Elig\_Rslt to A (Eligible)

**Else**

(1) Change value of Elig\_Rslt to B (Not Eligible)

In practice, Rulesets vary in complexity. To manage this complexity, rules can draw on different Expressions based on a wide range of Expression Types including Decision Tables. Decision tables can be downloaded into excel, edited and uploaded again into BRFplus. BRFplus also supports many other expression types: such as those to help implement mathematical calculations (Formula expression) or analyze tabular data (Loop expression, Table Operation expression).

The second example of a Ruleset used in DSM processing is a little more complex. This is an example of an eligibility check on a DSM Agreement. After the customer's application for the DSM program has been accepted as eligible, he or she then signs a DSM Agreement which outlines the requirements to qualify for the DSM Program incentives. In this example the Ruleset checks whether the customer has met the program goals and returns an eligibility flag accordingly. The program requirements are defined in a questionnaire as part of the DSM application object in CRM (refer to screenshot below). In the BRFplus function the

questionnaire is defined as a table of questions with components QUESTION\_ID, ANSWER\_ID, and VALUE (Data Dictionary element CRMT\_IU\_DSM\_BRF\_QUESTIONNAIRE).

The screen shot below shows an example questionnaire defined in a DSM Agreement in CRM.

The screenshot displays the SAP Energy Demand-Side Management for Utilities interface. The main title is "SAP Energy Demand-Side Management for Utilities". The breadcrumb trail shows "DSM Agreement: 6000003376 - Questionnaire - In Process". A "Back" button is visible. The left sidebar contains a navigation menu with the following items: Home, Worklist, Calendar, E-Mail Inbox, Activities, Account Management, Technical Data, Programs, Applications, and Agreements. Below the menu is a "Create" button with a sub-menu containing Program, Application, Agreement, and Survey. The main content area shows the "Agreement Eligibility" form. It includes a dropdown menu for "ID / Description" set to "AGREEMENT\_ELIGIBILITY" and a "Version" field. The form contains two questions:

**1) Does the new equipment match the measure exactly?**

- Not Known
- Yes
- No

**2) Is the new equipment installed and successfully tested?**

- Not Known
- Yes
- No

At the bottom of the form are "Save" and "Reset" buttons.

The next screen shot shows the data structure of the equivalent questionnaire defined in a BRFplus Data Object of type 'Table'.

**Table: QUESTIONNAIRE\_DATA, Table of questionnaire data**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

**Define Data Binding**

DDIC Binding: Bind to DDIC Table (Data Dict) Refresh Binding

DDIC Type Name: CRMT\_IU\_DSM\_BRF\_QUESTIONNAIRE

**Table Properties**

Table Line Type: Line of questionnaire data

**Components**

Component Name	Text	Type
QUESTION_ID		Element (Text)
ANSWER_ID		Element (Text)
VALUE		Element (Text)

The questionnaire table is then added in the signature tab as part of the definition of the 'DSM Agreement Eligibility Check' Function along with structures DSM\_MEASURE and DSM\_MEASURE\_CONFIGURATION. All three components will be used as context data in the Rulesets that will determine the eligibility of an Agreement.

**Function: CRM\_DSM\_CHECK\_AGR\_ELIG\_ENH1, Eligibility DSM Agr.**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Start Simulation

Properties | **Signature** | Assigned Rulesets

**Context**

Component Name	Text	Type
DSM_MEASURE	Measure Data	Structure
DSM_MEASURE_CONFIGURATION	Measure Config.	Structure
WATTS_BEFORE	Watts Before	Element (Number)
WATTS_AFTER	Watts After	Element (Number)
EVALUATION_RELEVANT	Meas.conf.relevant	Element (Text)
QUESTIONNAIRE_DATA	Table of questionnaire data	Table
CRMS_IU_DSM_BRF_QUESTIONNR_LINE	Line of questionnaire data	Structure
QUESTION_ID		Element (Text)
ANSWER_ID		Element (Text)
VALUE		Element (Text)

**Result Data Object**

Data Object: Eligibility result structure with multiple Rejection mes...

In the example, to be eligible for the program incentives the customer must either install energy efficient lighting or install and have tested an energy efficient refrigerator. To implement this logic in BRFplus the rule is defined at multiple levels. The first level of business logic states that if the Measure of Watts is not provided in the agreement then use the answers in the questionnaire to determine Eligibility:

**Ruleset: AGR\_ELIGIBILITY\_RULESET\_MAIN, Eligibility Agr main**

Hide Ruleset Header | Variables | Context Overview

Enabled:

Function: [Eligibility DSM Agr.](#)

Precondition: [<Not assigned>](#)

---

**Rules**

(1) Rule: [If header or measure](#) - Unlimited Validity

**If**

[Measure ...-Measure ID](#) is initial

**Then**

(1) Process Rule [Agr Header rule 1](#)

(2) Process Rule [Agr Header rule 2](#)

(3) Process Rule [Agr Header rule 3](#)

**Else**

(1) Change value of [Eligibil...-Elig.Rslt](#) to A (Eligible)

(2) Process Rule [Measure rule](#)

At the next level of logic the eligibility rules for questionnaire answers and wattage measurements are defined.

The wattage measurement rule is defined using two sub-rules. The business logic is as follows:

Not eligible when WATTS BEFORE < 40W with Reason 'Watts Before < 40 watts.'

Not eligible when WATTS AFTER >= WATTS BEFORE with Reason 'Watts After >= Watts Before'

**Rule: MEASURE\_ELIGIBILITY\_RULE, Measure rule**

Back | Edit | Check | Save | Activate | Delete | More

---

**General**

---

**Detail**

Context Overview | Start Simulation

**If**

[Measure ...-Meas.conf.relevant](#) is equal to X (true)

**Then**

(1) Process Rule [Measure rule 1](#)

(2) Process Rule [Measure rule 2](#)

**Rule: MEASURE\_ELIGIBILITY\_RULE1, Measure rule 1**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Context Overview | Start Simulation

**If**

Measure ...-Watts Before is less than 40

**Then**

(1) Change value of Eligibil...-Elig.Rslt to Not Eligible

(2) Change value of Eligibil...-Elig.BusRs1 to Watts before is less th

**Rule: MEASURE\_ELIGIBILITY\_RULE2, Measure rule 2**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Context Overview | Start Simulation

**If**

Measure ...-Watts After is greater than or equal to Measure ...-Watts Before

**Then**

(1) Change value of Eligibil...-Elig.Rslt to Not Eligible

(2) Change value of Eligibil...-Elig.BusRs2 to Watts after is >= Watts

The business logic to determine eligibility based on the questionnaire answers is defined using a series of three sub-rules. The first, if all answers to the questionnaire are YES then return result 'Eligible':

**Rule: AGR\_HEADER\_ELIGIBILITY\_RULE1, Agr Header rule 1**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Context Overview | Start Simulation

**If** [All answers are yes](#)

**Then**

(1) Change value of [Eligibil...-Eliq.Rslt](#) to  Eligible

The second and third rules use table operations to implement the logic. The Agreement is not eligible if the answer to the first question is NO with reason 'Equipment does not match measures'. The Agreement is also not eligible if the answer to the second question is NO with reason 'Equipment is not installed or measured'. To drill down to the table operation select the hyperlink 'First Answer is Yes' as shown in the screen shot below.

**Rule: AGR\_HEADER\_ELIGIBILITY\_RULE2, Agr Header rule 2**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Context Overview | Start Simulation

**If**

[First answer is YES](#) is equal to false

**Then**

(1) Change value of [Eligibil...-Eliq.Rslt](#) to  Not Eligible

(2) Change value of [Eligibil...-Eliq.BusRs1](#) to

**Table Operation: FIRST\_ANSWER, First answer is YES**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Context Overview | Start Simulation

If table **Table of questionnaire data** has at least  rows then return True in result **Boolean** else False

**With Selection Conditions:**

**QUESTION\_ID** is equal to  **And**

**VALUE** is equal to

**Rule: AGR\_HEADER\_ELIGIBILITY\_RULE3, Agr Header rule 3**

Back | Edit | Check | Save | Activate | Delete | More

**General**

**Detail**

Context Overview | Start Simulation

**If**

**Second answer is YES** is equal to false

**Then**

(1) Change value of **Eligibil...-Elig\_Rslt** to  Not Eligible

(2) Change value of **Eligibil...-Elig\_BusRs2** to

Finally all objects need to be activated and then the decision service is ready to be used.

## Decision Service Testing

A decision service can be tested directly in the BRFplus workbench on the Function user interface with the button Start Simulation. On the simulation screen values can be entered or loaded from excel.

**Simulation**

***Business Rule Framework plus - Simulation***

[← Back to Selection](#) | 
 [Execute](#) | 
 [Execute and Display Processing Steps](#)

**▼ Selected Object**

Function Name:

Description:

**▼ Context Values**    [Import Test Data](#)

**Measure Config.**

Watts Before:

Watts After:

Meas.conf.relevant:  X     unknown

**Measure Data**

Measure ID:

Measure Desc.:

Item Cat.:

Product:

When using the option “Execute and Display Processing Steps” the simulation output will be very detailed showing context, processing steps and result(s) in a well-organized execution hierarchy tree, which also can be downloaded into Microsoft Excel.

The screenshot shows the Simulation tool interface. The top section is titled "Context Values" and contains a "Result" section. Under "Result", there is a heading "Eligibility result structure with multiple Rejection messages". Below this heading, there are several input fields: "Elig.Rst:" with a dropdown menu showing "A" and the text "Eligible"; "Elig.BusRs1:" through "Elig.BusRs5:" each with a dropdown menu showing "Reb:".

The bottom section is titled "Processing Steps" and contains a table with the following columns: Step, Type, Status, and Value.

Step	Type	Status	Value
Trace for Eligibility DSM Agr. started on 02.02.2012 05:14:46 by user VOLKDI	Trace		
Eligibility DSM Agr.	Function	STARTED	
Event Processing			
Context			
Measure Data	Data Object		
Measure Config.	Data Object		
Table of questionnaire data	Data Object		
Eligibility Agr main	Ruleset	STARTED	
Rule processing: If header or measure (position 000001)			
If header or measure	Rule	STARTED	
If test parameter Measure ID (MEASURE_ID); value WATTS			
Test parameter is not initial			
Condition not fulfilled			

Finally, the decision service can also be tested in the ABAP debugger and in the UI.

## Decision Service Trigger Implementation

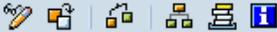
After creating and activating the data objects and functions, the new decision service is ready to be plugged into the right place in the process. Often you can find BAdI s, enhancement spots or other exit options for exactly that purpose.

The DSM eligibility check functions are called using actions in the Post Processing Framework in CRM. To implement this you define your coding in the BAdI for execution of actions (EXEC\_METHODCALL\_PPF). If you need to read and transfer data to your BRFplus function you also create a BAdI implementation for BAdI "Context for BRFplus Functions in DSM" (CRM\_IU\_DSM\_BRF\_CONTEXT\_FILLER).

If triggering actions and conditions for the DSM application and agreement eligibility checks are required other than those delivered as standard, then you also need to make your action profile configuration settings and assign your action profile to the DSM transaction type in the Configuration Workbench (SPRO). More information on these CRM configuration steps can be found in the Implementation Guide (IMG) documentation under CRM->Industry-Specific Solutions-> Utilities Industry -> Demand Side Management -> DSM Processing -> Actions in DSM Processing. Further general information on the Post Processing Framework can be found on the SAP help portal:

[http://help.sap.com/saphelp\\_crm700\\_ehp01/helpdata/en/de/a38338d22aa947e10000009b38f8cf/frameset.htm](http://help.sap.com/saphelp_crm700_ehp01/helpdata/en/de/a38338d22aa947e10000009b38f8cf/frameset.htm)

The screenshot below shows the BAdI implementation in transaction SE19.

**Business Add-In Builder: Display Implementation CRM\_DSM\_ELIGBLE\_CHK1**

 Definition Documenta Documentation
Implementation Name  ActiveImplementation Short Text Definition Name Runtime Behavior 

Properties Interface

Interface name Name of implementing class: 

Method	Implemen...	Description
EXECUTE	ABAP Co...	Execution of Processing Logic

Default implementation class Example implementation class **Class Builder: Class CL\_IM\_CRM\_DSM\_ELIGBLTY\_CHK Display**

 Pattern Pretty Printer Signature  Public Section  Protected Section  Private Section
Method  Active

```

1  method IF_EX_EXEC_METHODCALL_PPF-EXECUTE.
2
3      include crm_status_con.
4
5      extract_order_information( io_appl_object = io_appl_object ).
6
7      data lv_current_status type crm_j_status.
8      lv_current_status = get_status_of_order( ).
9
10     if lv_current_status <> gc_status-dsm_in_process. "In Process
11
12         log_initial_status_not_valid( iv_application_log = ip_application_log ).
13
14         rp_status = '2'. " Not successfull
15         return.
16
17     endif.
18
19     data lr_eligibility_exception type ref to cx_crm_iu_dsm_exception.
20     data lv_order_is_eligible type crmt_boolean value abap_false.
21     data lt_eligibility_check_log type bapiret2_tab. " The log of what happens during the eligibility check
22
23     try.
24
25         is_order_eligible( importing ev_is_eligible = lv_order_is_eligible changing ct_log = lt_eligibility_check_log ).
26
27     catch cx_crm_iu_dsm_exception into lr_eligibility_exception.

```

**Class Builder: Class CL\_IM\_CRM\_DSM\_ELIGBLTY\_CHK Display**

Method: IS\_ORDER\_ELIGIBLE Active

```

1  method is_order_eligible.
2
3  data lr_eligibility_checker type ref to cl_crm_iu_dsm_eligibility_chk.
4  create object lr_eligibility_checker exporting iv_order_guid = gv_order_guid iv_process_id = gs_process_information-process_id.
5
6  lr_eligibility_checker->is_order_eligible( importing ev_is_eligible = ev_is_eligible changing ct_log = ct_log ).
7
8  endmethod.

```

**Class Builder: Class CL\_CRM\_IU\_DSM\_ELIGIBILITY\_CHK Display**

Method: IS\_ORDER\_ELIGIBLE Active

```

1  method is_order_eligible.
2
3  " Retrieve the BRF+ function name from the program for eligibility check
4  data lv_BRF_function_ID type fdt_uuid.
5  retrieve_BRF_function_id( importing ev_function_id = lv_brf_function_id
6                          changing ct_log = ct_log ).
7
8  " Get the result from the BRF function module.
9  data lr_eligibility_result type ref to if_fdt_result.
10 execute_BRF_function( exporting iv_function_ID = lv_BRF_function_ID
11                      importing er_function_result = lr_eligibility_result
12                          changing ct_log = ct_log ).
13
14 " Interpret the result of the BRF function module
15 evaluate_function_result( exporting iv_function_id = lv_BRF_function_ID
16                          ir_function_result = lr_eligibility_result
17                          importing ev_is_eligible = ev_is_eligible
18                          changing ct_log = ct_log ).
19
20 log_eligibility_outcome( exporting iv_is_eligible = ev_is_eligible changing ct_log = ct_log ).
21
22 endmethod.

```

In the code above an instance of the BRFplus function is created. Note that the ID can be found in the BRFplus workbench in Function definition screen. The BRF function is then executed to return the function result.

The filling of context values i.e. the import parameters happens in the example BAdI implementation below:

**Enhancement Implementation CRM\_IU\_DSM\_ELIGIBILITY\_CHECK Display**

Enhancement Implementation: CRM\_IU\_DSM\_ELIGIBILITY\_CHECK Active

Properties | History | Technical Details | **Enh. Implementation Elements**

BAdI Implementations	Description
CRM_IU_DSM_ELIGIBI	Context Filler for BRFplus
• Implementing Class	
• Filter Val.	
CRM_IU_DSM_QUESTI	Fills out the context for D

Implementing Class	
Interface	IF_CRM_IU_DSM_BRF_CONTEXT_FILL
Implementing Class	CL_IM_CRM_DSM_APL_ELIG_CONTEXT
Method	Short Description
IF_CRM_IU_DSM_BRF_CONTEXT_FILL~FILL_OUT_CONT...	Fills out the context of a BRF+ Funct...
GET_APPLICATION_GUID	Gets the application guid from the p...

```

IF_CRM_IU_DSM_BRF_CONTEXT_FILL~FILL_OUT_CONTEXT | Active
method IF_CRM_IU_DSM_BRF_CONTEXT_FILL~FILL_OUT_CONTEXT.

" These constants are the names of the data objects within the context of the BRF+ function
data gc_brif_application_header type if_fdt_types=>name value 'DSM_APPLICATION_HEADER'.
data gc_brif_partner_data type if_fdt_types=>name value 'CUSTOMER_DATA'.
data gc_brif_premise_data type if_fdt_types=>name value 'PREMISE_DATA'.

data lv_application_guid type crmt_object_guid.
gt_log = ct_log.
lv_application_guid = get_application_guid( it_context_parameters ).

data ls_header_data type crms_iu_dsm_brif_appl_header.
cl_crm_iu_dsm_brif_data_providr=>read_dsm_application_header( exporting iv_application_guid = lv_application_guid
importing es_dsm_application_header = ls_header_data ).
cr_context->set_value( exporting iv_name = gc_brif_application_header ia_value = ls_header_data ).

data ls_customer_data type crms_iu_dsm_brif_partner_data.
cl_crm_iu_dsm_brif_data_providr=>read_customer_data_of_app( exporting iv_application_guid = lv_application_guid
importing es_customer_data = ls_customer_data ).
cr_context->set_value( exporting iv_name = gc_brif_partner_data ia_value = ls_customer_data ).

data ls_premise_data type crms_iu_dsm_brif_premise_data.
cl_crm_iu_dsm_brif_data_providr=>read_premise_details( exporting iv_application_guid = lv_application_guid
importing es_premise_details = ls_premise_data ).
cr_context->set_value( exporting iv_name = gc_brif_premise_data ia_value = ls_premise_data ).

endmethod.

```

## The End Result

The screenshot above shows the end result of the BRFplus DSM Agreement Eligibility Check Function called by executing an Action from the DSM Agreement in the CRM Utilities Solution.

The screenshot displays the SAP Energy Demand-Side Management for Utilities interface. The main window shows a 'DSM Agreement: 6000003376 - Rejected' with a status of 'Rejected'. A yellow tooltip provides details: 'Eligibility check was not successful; equipment is not installed or tested', 'Measures that were checked are eligible', and 'DSM Agreement is not eligible and is now rejected'. Below the tooltip, the 'Actions' section shows a table of actions:

Actions	Status	Action Definition	P...	Cr...	St...	Cr...	Cre...	Exe...
	●	MANU_AGR_ELIGIBILITY_CHECK1	M...	V...	ln...	1...	03...	Done
	●	MANU_AGR_ELIGIBILITY_CHECK1	M...	V...	ln...	1...	03...	Done
	■	MANU_AGR_ELIGIBILITY_CHECK1	M...	V...	Pr...	1...	03...	Done

The 'Questionnaire' section below shows a table with columns for ID, Description, and Version:

A...	ID	Description	Version
AGREEMENT_ELI...	desc.		000000001 15.12.2011 03:25:18 VOLKDI

With an effort of just a few hours BRFplus can be used to implement a decision service to cater for automation of many tasks, freeing up time for more value adding work. The business rules that implement the decision service can be understood and even be changed by the domain expert so that the IT professional does not need to understand the business details. Changes to the business rule logic can be implemented and tested without the need to change the underlying application code.

## Related Content

For more information, visit the [Business Rules Management homepage](#).

## Copyright

© Copyright 2011 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Oracle Corporation.

JavaScript is a registered trademark of Oracle Corporation, used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects S.A. in the United States and in other countries. Business Objects is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

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