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

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:

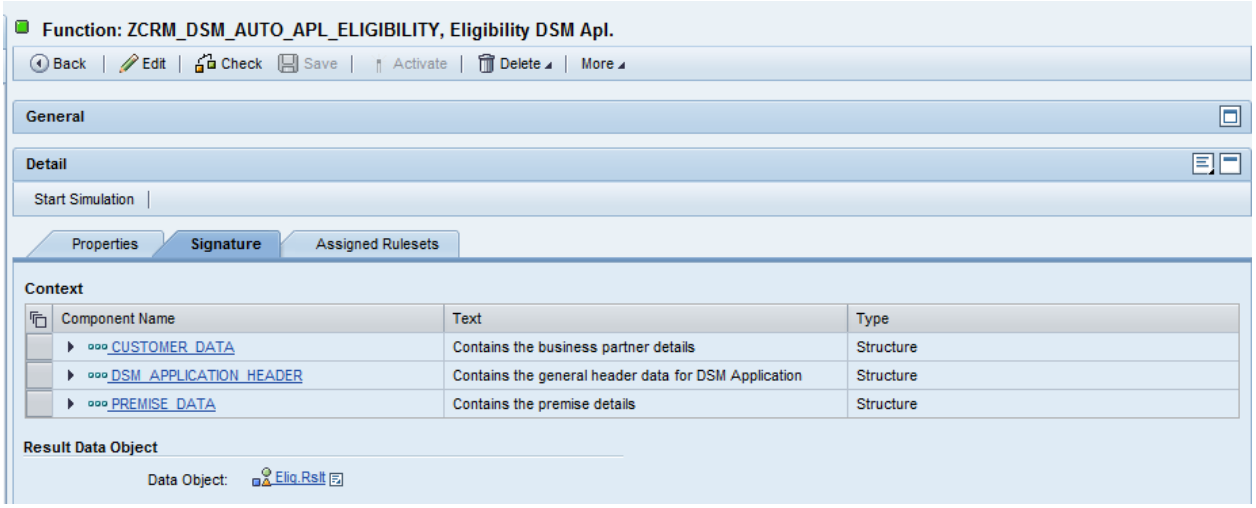
- Name "DSM Eligibility Check"
- Description "Checks the eligibility of a DSM Application"

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.

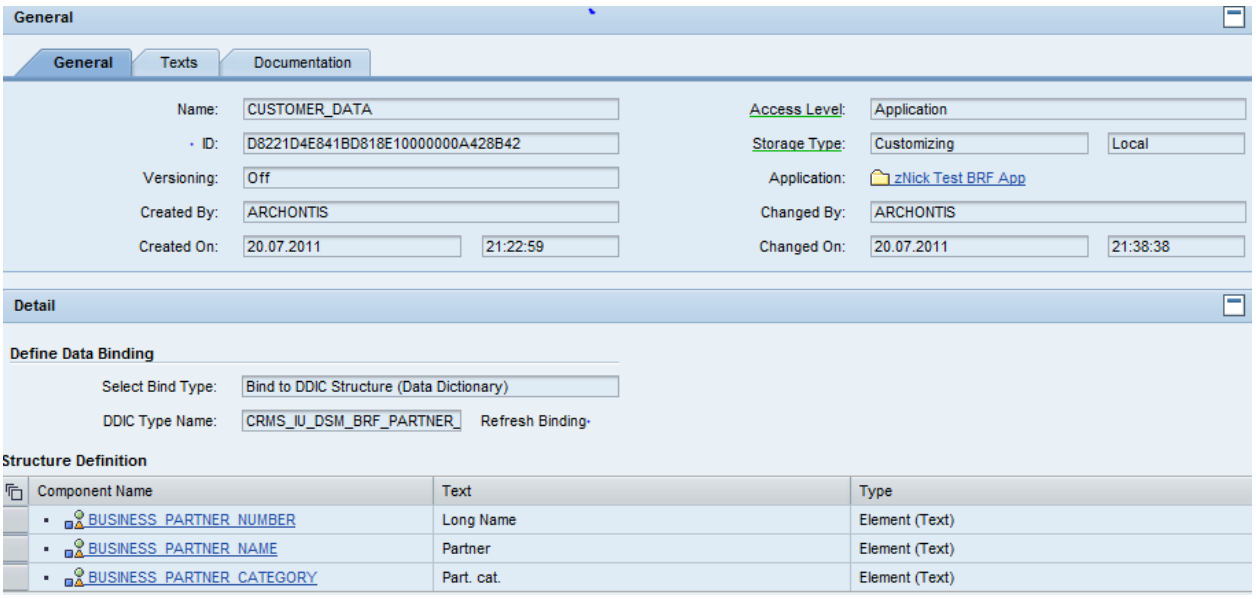
- Inputs/Context
  - Customer Data Structure (CRMS_IU_DSM_BRF_PARTNER_DATA)
  - DSM Application Header (CRMS_IU_DSM_BRF_APPL_HEADER)
  - Premise Data (CRMS_IU_DSM_BRF_PREMISE_DATA)
- Outputs/Results
  - Eligibility Result (IS_ELIGIBLE)

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+).



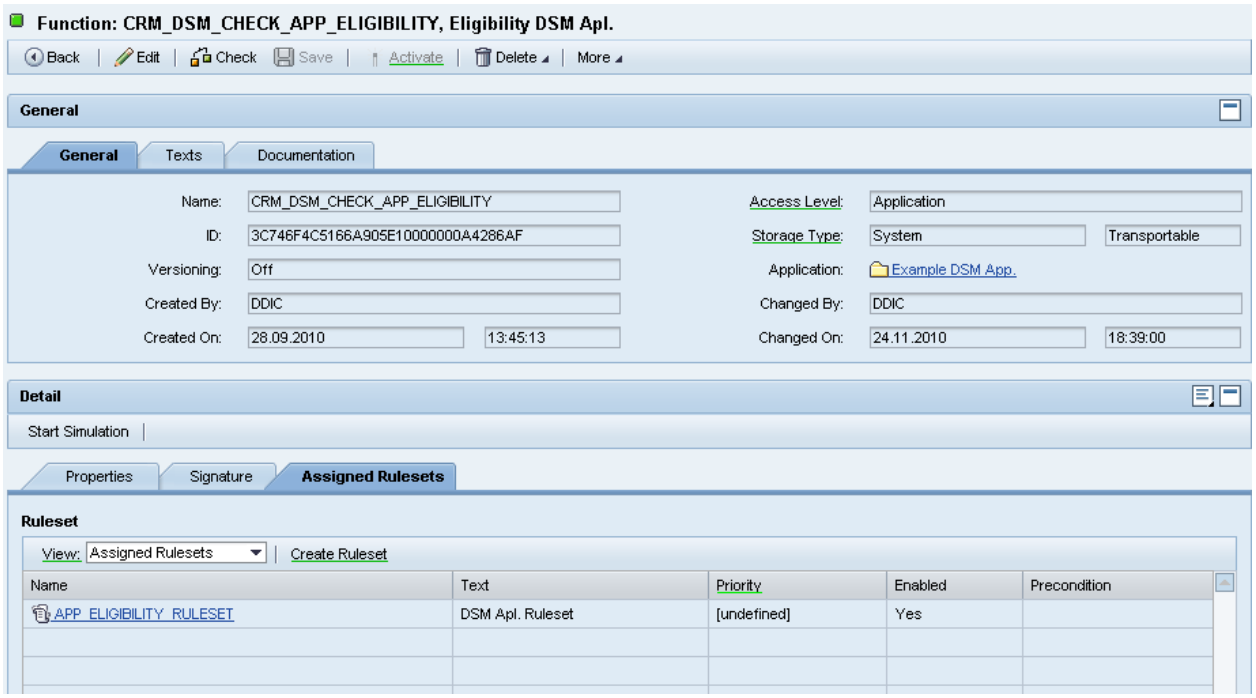
Data objects are used in BRFplus to define parameters. The parameter Customer Data, for example, is defined by a data object shown in the next screenshot. This can be reached by selecting the hyperlink on CUSTOMER_DATA in the Signature tab. You can see that properties such as texts, element type (text), 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.



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.



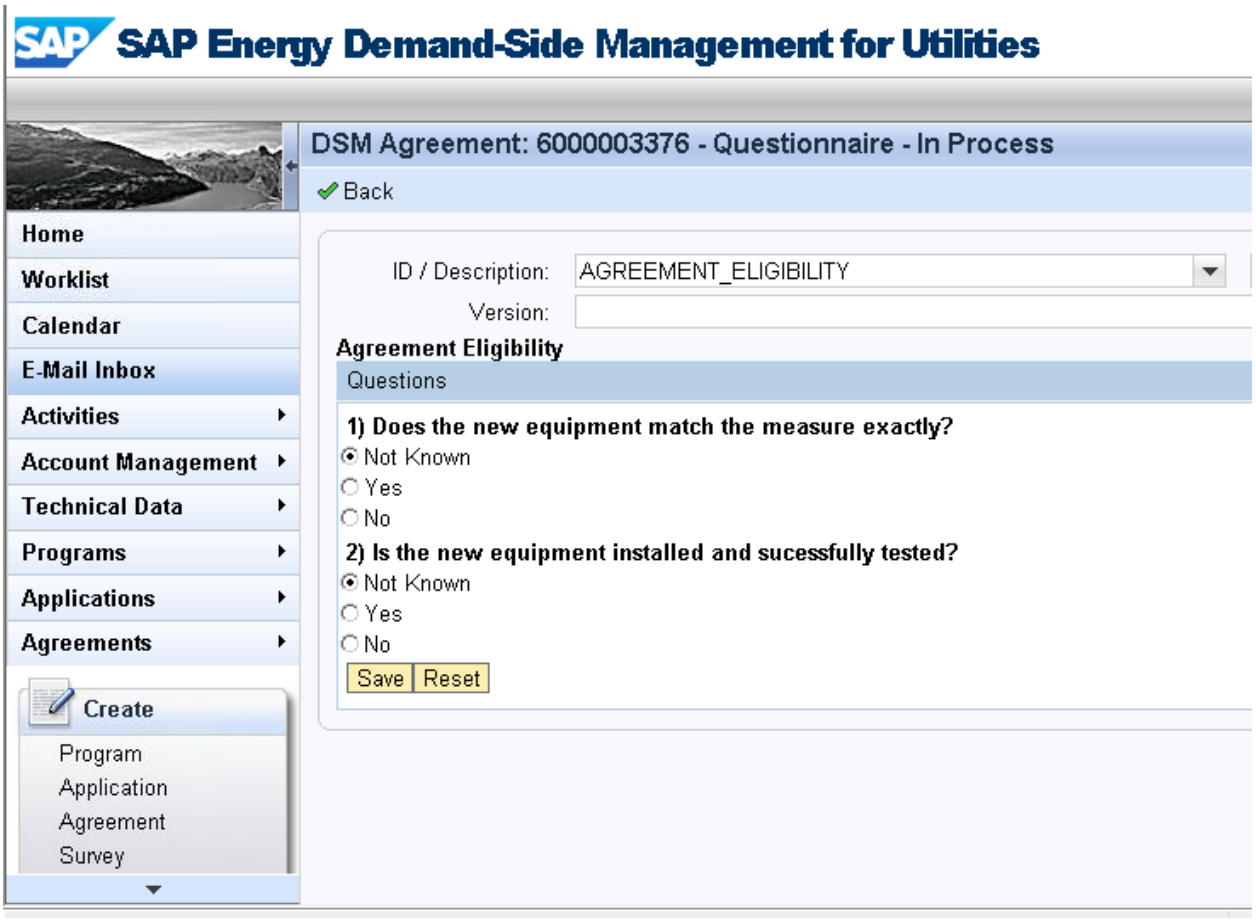
The Ruleset (below) is an example of a very simple rule. The rule states that if the Business Partner Category is of type 'Person' (i.e. not a company) then the result is "Eligible' else the Business Partner is 'Ineligible' to participate in the DSM program.

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).

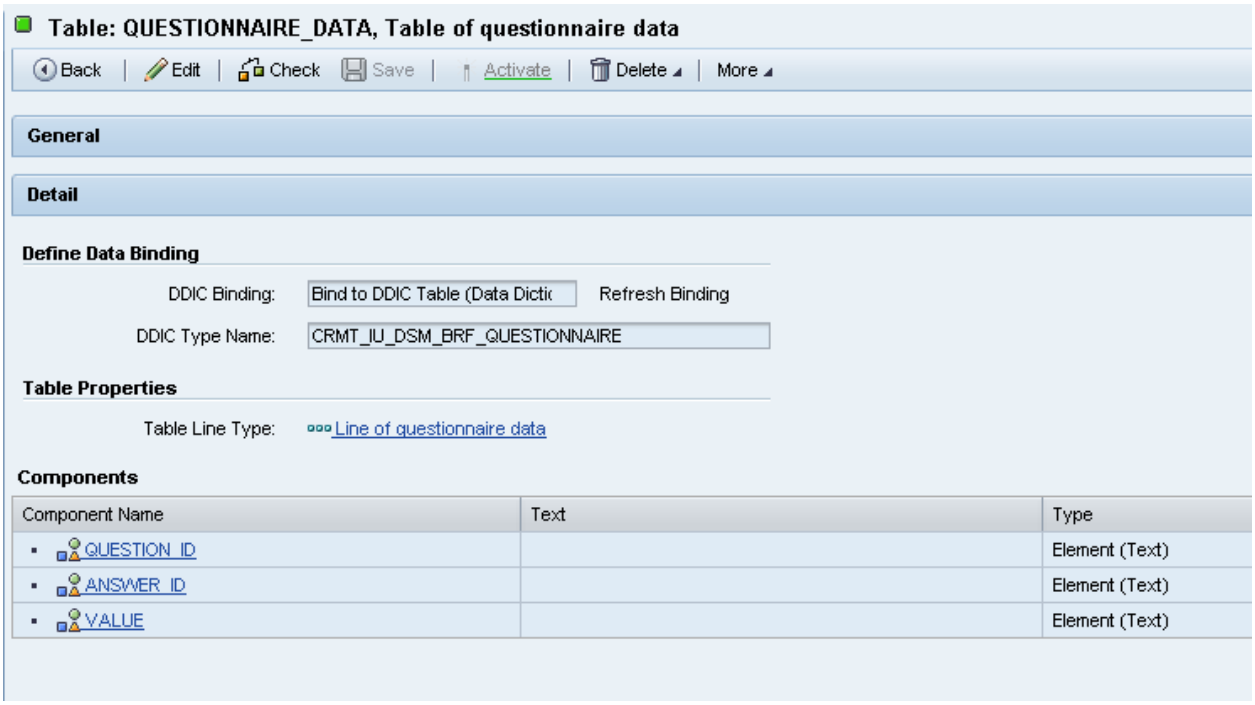
The second example of a Ruleset used in DSM processing is a little more complex. This is an example of an eligibility check on a DSM Agreement. After the customer's application for the DSM program has been accepted as eligible, he or she then signs a DSM Agreement which outlines the requirements to qualify 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).

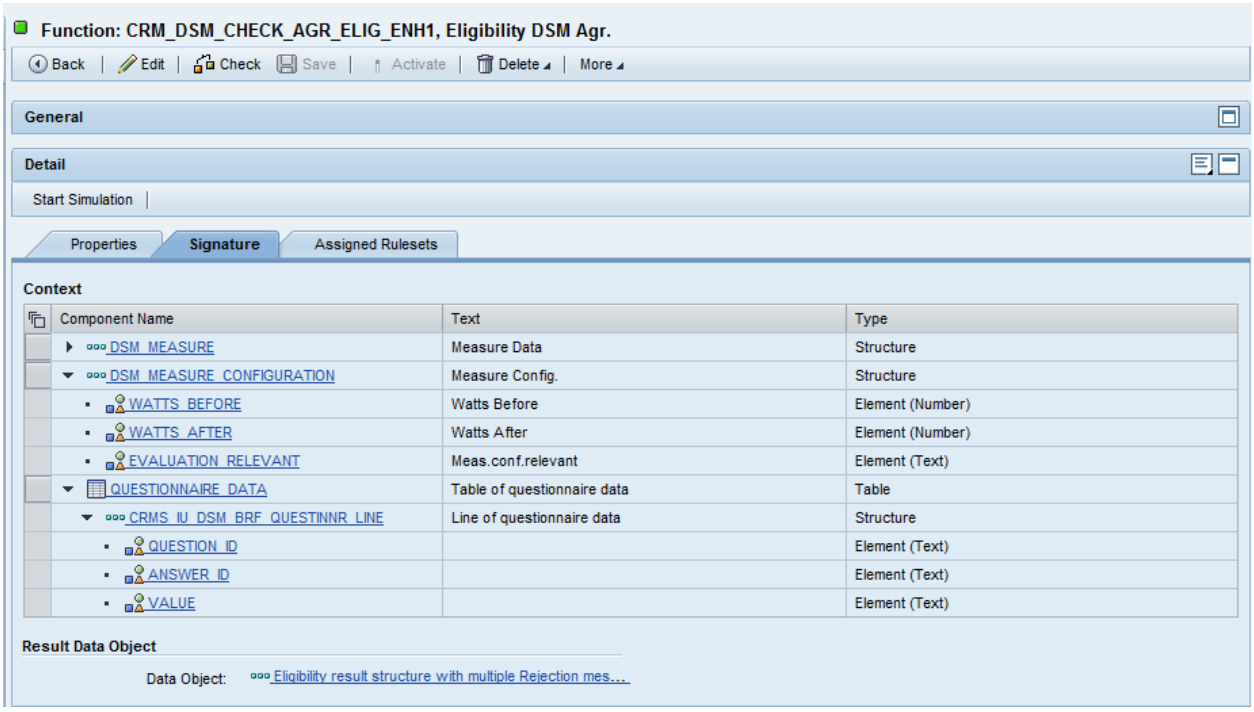
The screen shot below shows an example questionnaire defined in a DSM Agreement in CRM.



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



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.



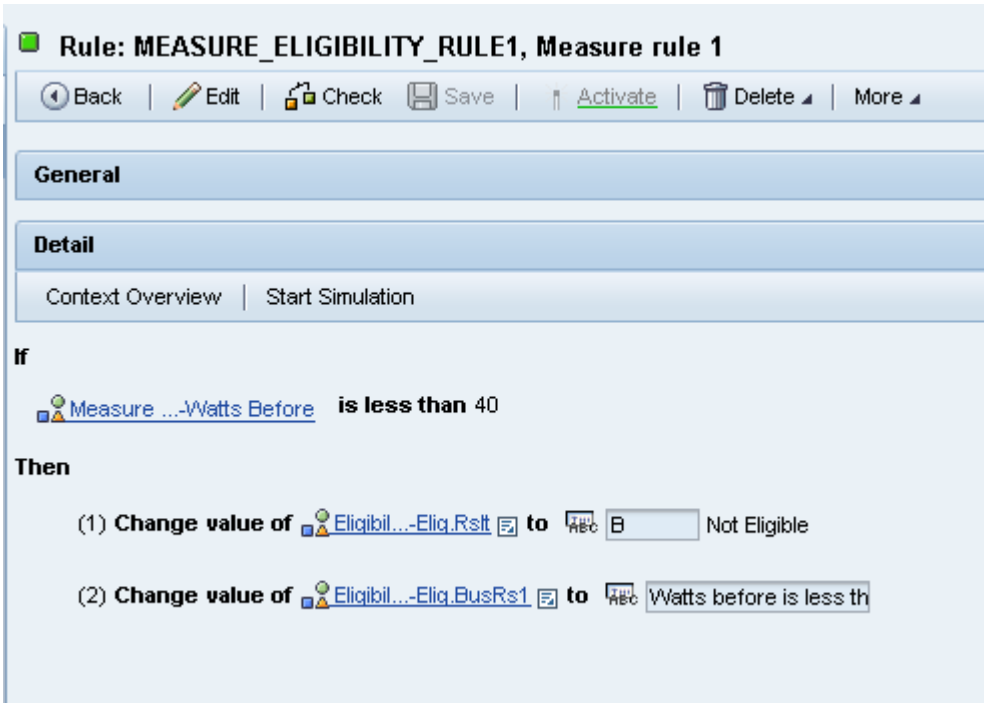
In the example, to be eligible for the program incentives the customer must either install energy efficient lighting or install and have tested an energy efficient refrigerator. To implement this logic in BRFplus the rule is defined at multiple levels. The first level of business logic states that if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

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:

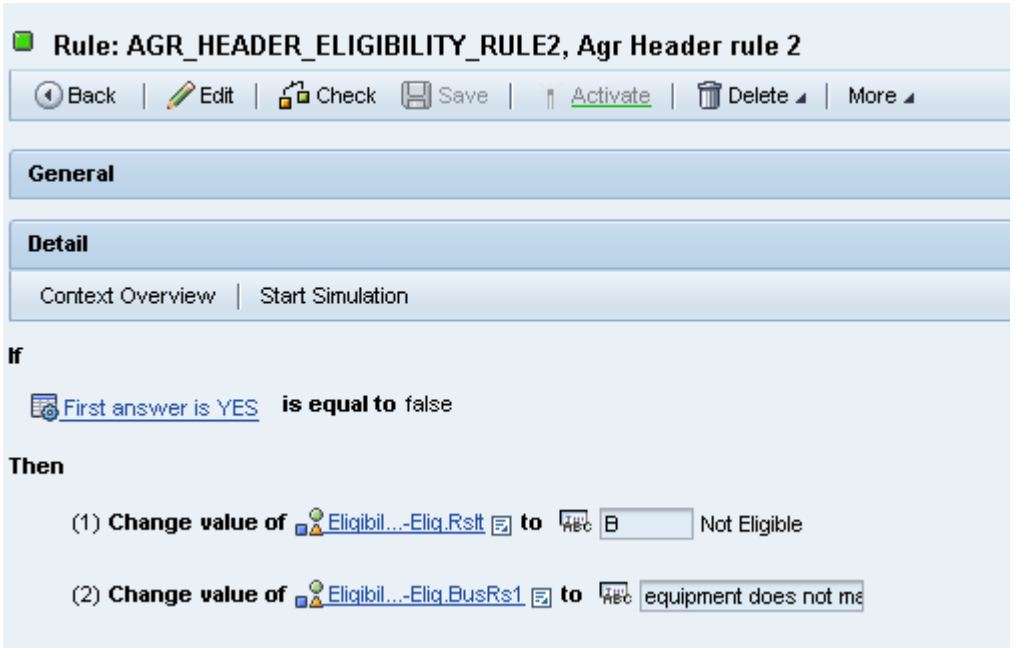
Not eligible when WATTS BEFORE < 40W with Reason ‘Watts Before < 40 watts.’

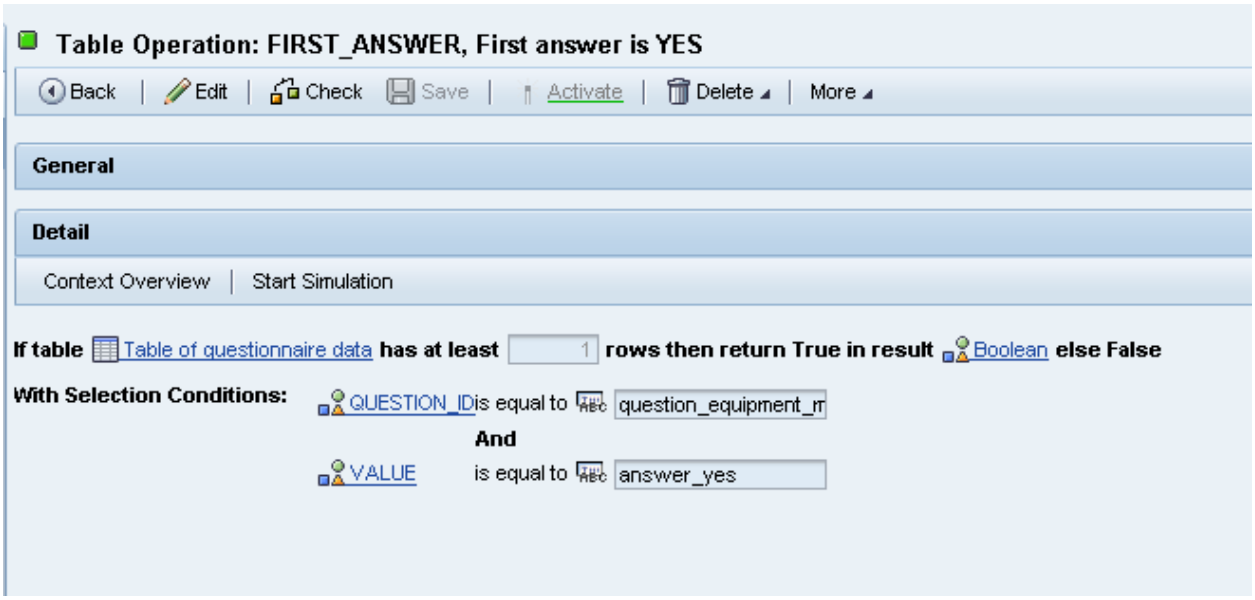
Not eligible when WATTS AFTER >= WATTS BEFORE with Reason ‘Watts After >= Watts Before’



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’:

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.

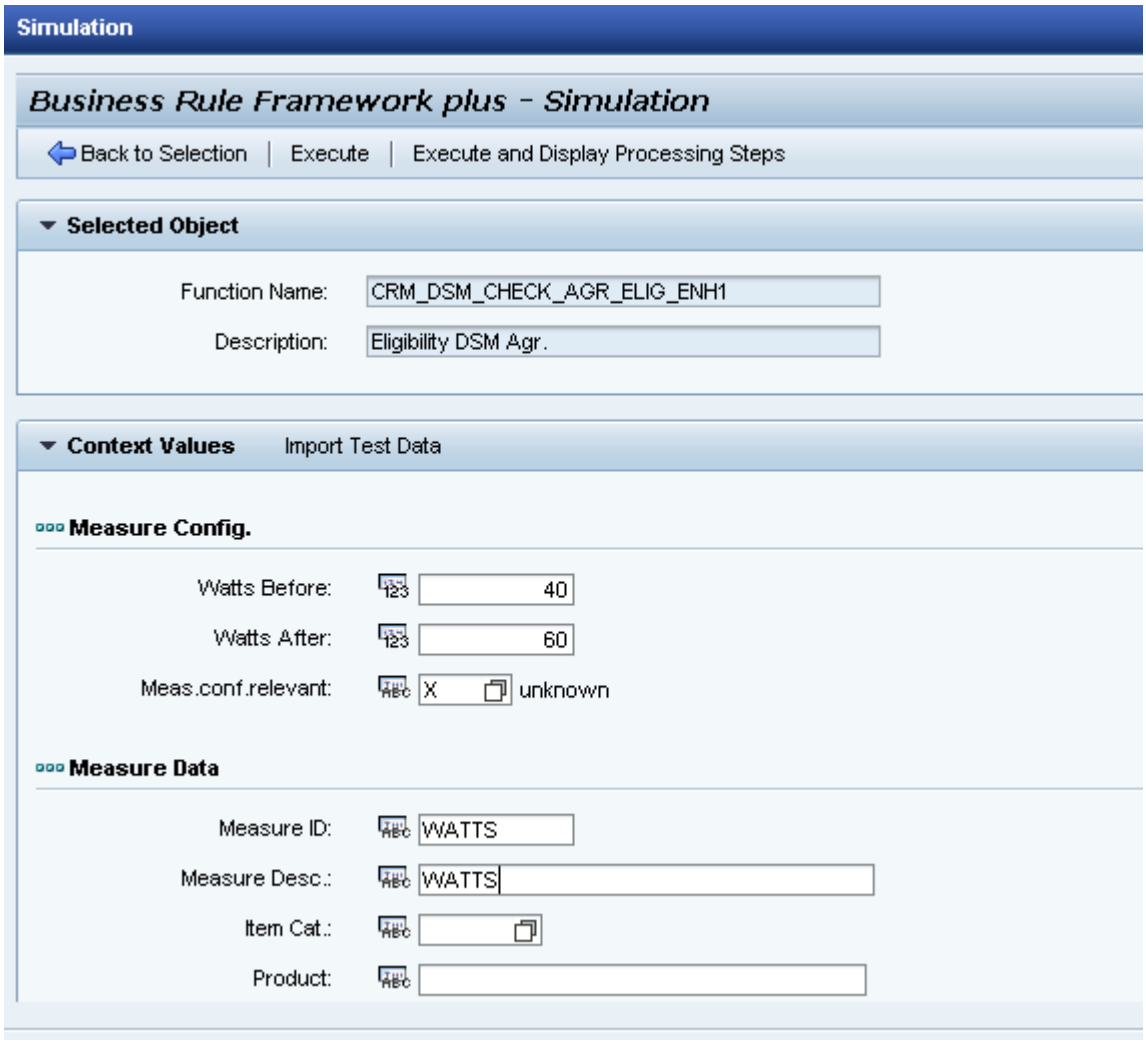




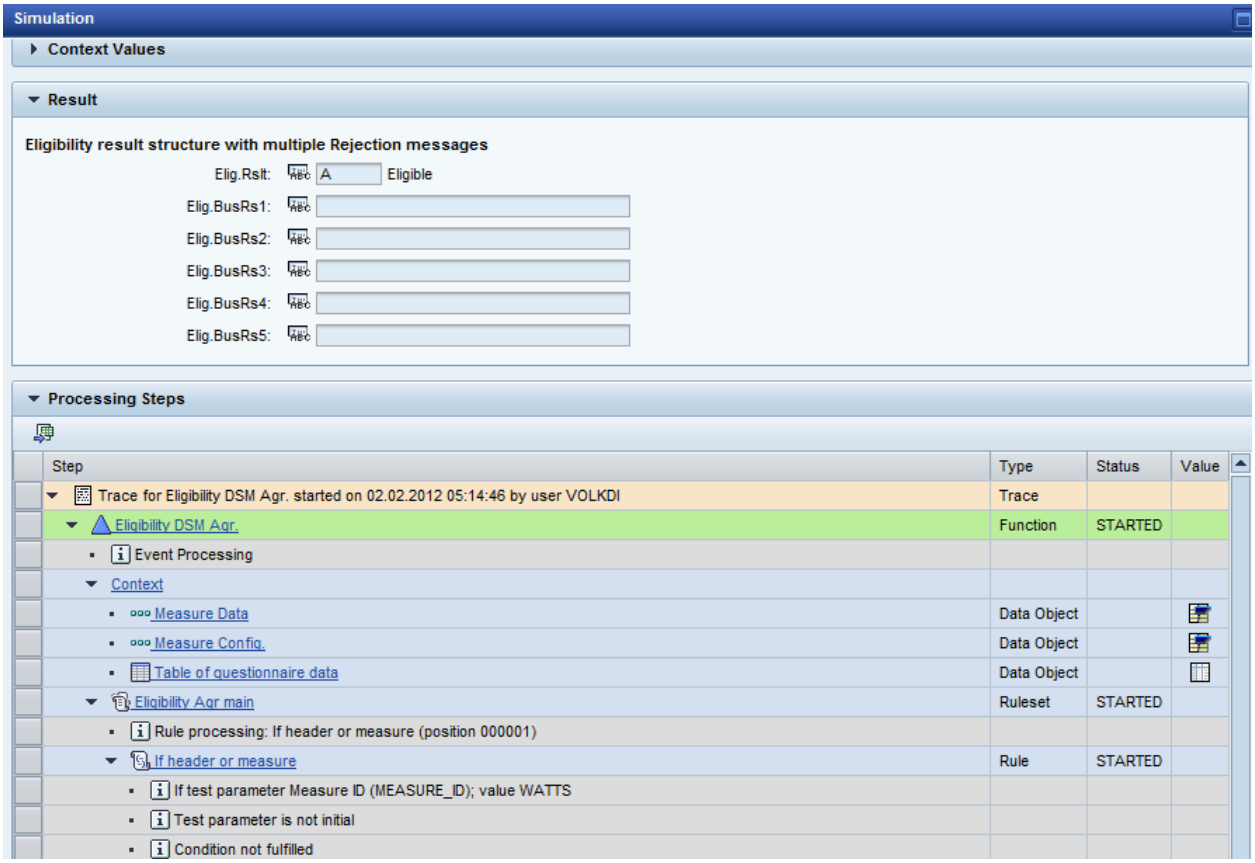
Finally all objects need to be activated and then the decision service is ready to be used.

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.



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.



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

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	CRM_DSM_ELIGBLE_CHK1 Active
Implementation Short Text	Eligibility check Implementation for DSM action on header
Definition Name	EXEC_METHODCALL_PPF
Runtime Behavior	Implementation will be called

Properties | Interface

Interface name	IF_EX_EXEC_METHODCALL_PPF
Name of implementing class:	CL_IM_CRM_DSM_ELIGIBILITY_HEAD

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

Default implementation class

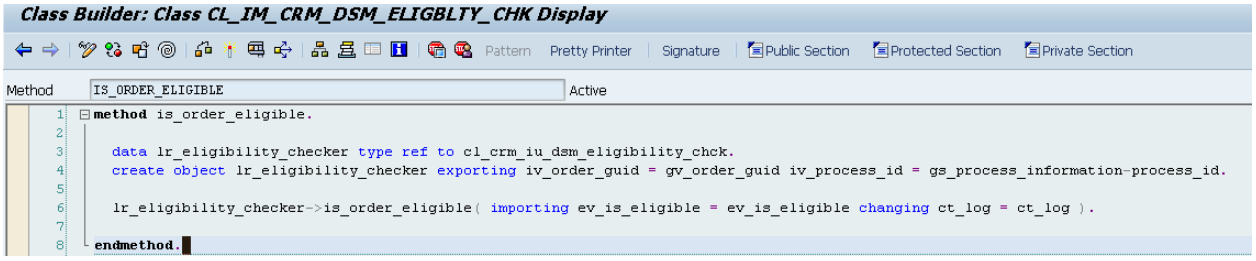
Example implementation class

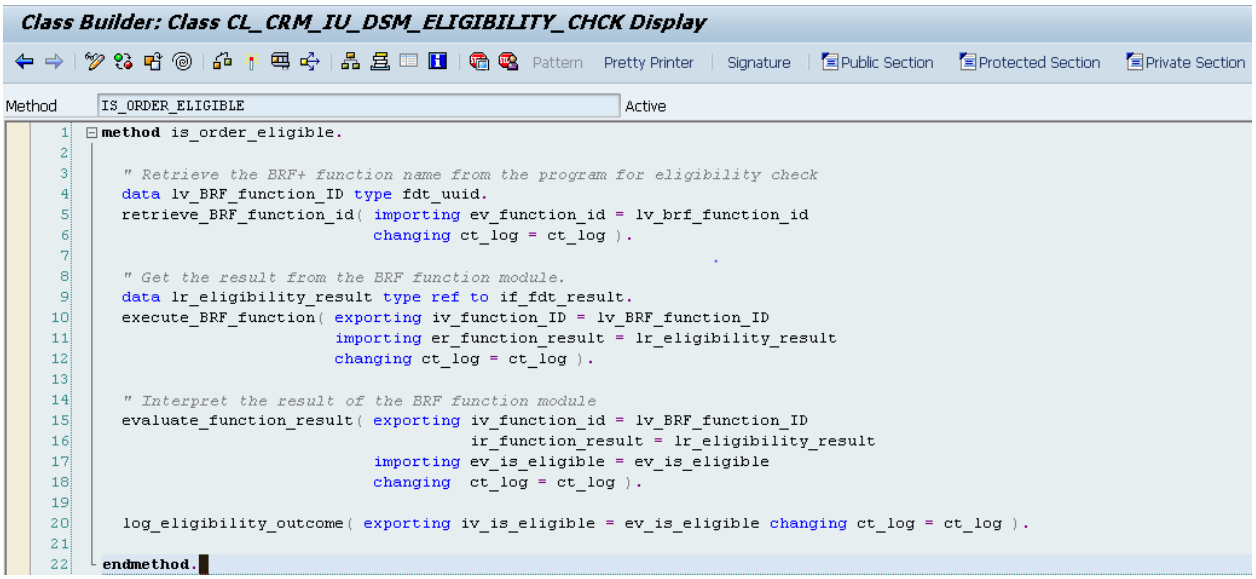
Class Builder: Class CL_IM_CRM_DSM_ELIGBLTY_CHK Display

Pattern | Pretty Printer | Signature | Public Section | Protected Section | Private Section

Method IF_EX_EXEC_METHODCALL_PPF~EXECUTE Active

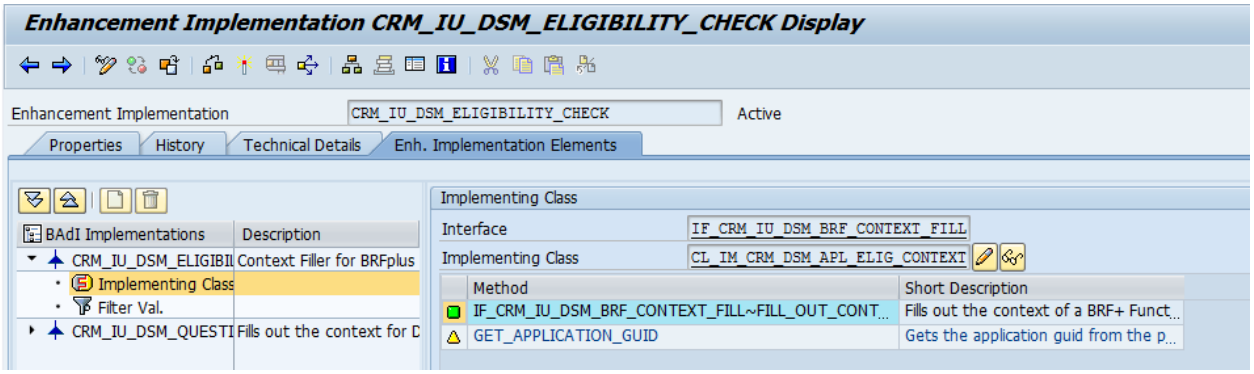
```
method IF_EX_EXEC_METHODCALL_PPF~EXECUTE.

  include crm_status_con.

  extract_order_information( io_appl_object = io_appl_object ).

  data lv_current_status type crm_j_status.
  lv_current_status = get_status_of_order( ).

  if lv_current_status <> gc_status-dsm_in_process. "In Process

    log_initial_status_not_valid( iv_application_log = ip_application_log ).

    rp_status = '2'. " Not succesfull
    return.

  endif.

  data lr_eligibility_exception type ref to cx_crm_iu_dsm_exception.
  data lv_order_is_eligible type crmt_boolean value abap_false.
  data lt_eligibility_check_log type bapiret2_tab. " The log of what happens during the eligibility check

  try.

    is_order_eligible( importing ev_is_eligible = lv_order_is_eligible changing ct_log = lt_eligibility_check_log ).

  catch cx_crm_iu_dsm_exception into lr_eligibility_exception.
```





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.

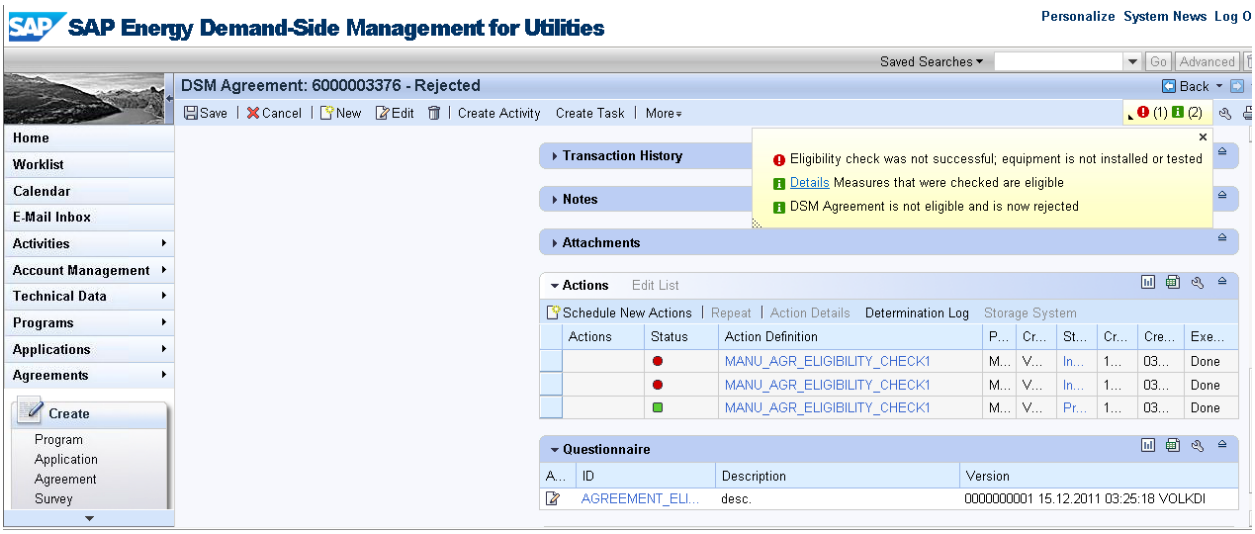
The filling of context values i.e. the import parameters happens in the example BAdI implementation below:



```
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_brf_application_header type if_fdt_types=>name value 'DSM_APPLICATION_HEADER'.
  data gc_brf_partner_data type if_fdt_types=>name value 'CUSTOMER_DATA'.
  data gc_brf_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_brf_appl_header.
  cl_crm_iu_dsm_brf_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_brf_application_header ia_value = ls_header_data ).


  data ls_customer_data type crms_iu_dsm_brf_partner_data.
  cl_crm_iu_dsm_brf_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_brf_partner_data  ia_value = ls_customer_data ).


  data ls_premise_data type crms_iu_dsm_brf_premise_data.
  cl_crm_iu_dsm_brf_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_brf_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.



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