

SAP How-To Guide for MDG-F

How to Read Approval Info for Master Data by Calling MDG API



Applies to

Master Data Governance for Financials (MDG-F) with release version 6.0 and newer. For more information, visit the Master Data Management homepage (<https://go.sap.com/community/topic/master-data-governance.html>).

Summary

SAP Master Data Governance provides out-of-the-box solutions for the central management of master data objects. Domain-specific solutions include business partner (MDG-BP), customer (MDG-C), supplier (MDG-S), material (MDG-M), and financials governance (MDG-F).

This document applies for all MDG master data. It is especially useful for the G/L Account because of the SOX (Sarbanes-Oxley Act) compliance.

In the G/L Account area, MDG-F is also known for its SOX compliance. SOX requires thorough tracking of changes with approval processes. This document shows you how to get relevant approval information for the G/L Account by calling all MDG APIs.

Author(s): Kefeng Wang

Company: SAP SE

Created on: October 2016

Version: 1.1

Table of Contents

Applies to	1
Summary.....	1
Business Scenario	3
Implementation	4
1. Read G/L Account Information	4
1.1. G/L Account Definition	4
1.2. Search Account	5
2. Read Change Request and Approval Information.....	6
2.1. Read Change Request by G/L Account Key	6
2.2. Read Workflow Log by Change Request	7
Additional Information	9
Links	9
How-to Guides	9
SAP Notes.....	9
Version History	9
Copyright.....	10

Business Scenario

SAP Master Data Governance (MDG) is used in Master Data Management (MDM), that is, the centralized, out-of-the-box, domain-specific creation, modification, and distribution of master data with a focus on SAP Business Suite.

Domain-specific content (data models, user interfaces, workflows) is part of the standard delivery for several application areas. It is a common requirement from customers to adapt the MDG data models to their specific needs.

The processes are workflow-driven and can include several approval and revision phases with the collaboration of all users participating in the master data processing.

In the G/L Account area, MDG-F is also known for its SOX (Sarbanes-Oxley Act) compliance. Customers implement MDG-F to fulfill the SOX requirement.

In MDG-F, the end user can navigate to a change document from the object search result list. From the change document he can then navigate to a change request and from there he can further navigate to the approval log (workflow log). For these standard functions, there are steps needed to get the approval information for master data.

If you want to create a custom UI to list audit info by object, this document provides all necessary APIs and sample code how to call them using the example of G/L Account.

Additionally, all these APIs are valid for all MDG master data. If you are required to do the same for other MDG objects, you only have to replace G/L Account with your object.

Implementation

Prerequisite: You are using standard MDG-F data model 0G for your G/L Account management, and standard/extended MDG workflow or rule based workflow.

1. Read G/L Account Information

1.1. G/L Account Definition

To get the correct G/L Account, you need a search screen to enter the search criteria. Therefore, you need a structure to define the search criteria or any account attribute to be listed in the search result list. With the standard data model 0G, there are several generated structures that can be used for your UI definition:

/MDG/_S_0G_PP_ACCOUNT: Chart of Account data

/MDG/_S_0G_PP_ACCCCDET: Company Code data

The following screenshot shows the MDG 7.0 data structure for Chart of Account data. In lower releases there will probably be fewer fields and in higher releases more fields, but this should not make a big difference for your usage.

Component	Typing Method	Component Type	Data Type	Length	Deci...	Short Description
COA	Types	KTOPL	CHAR	4	0	Chart of Accounts
ACCOUNT	Types	USMDZ1_ACCOUNT	CHAR	10	0	Account
ACCBLCREA	Types	USMDZ1_XSPEA	CHAR	1	0	Indicator: account is blocked for creation ?
ACCBLPLAN	Types	USMDZ1_XSPEP	CHAR	1	0	Indicator: account blocked for planning ?
ACCBPOST	Types	USMDZ1_XSPEB	CHAR	1	0	Indicator: Is Account Blocked for Posting?
ACCDEL	Types	USMDZ1_XLOEV	CHAR	1	0	Indicator: Account marked for deletion?
ACCGRPACC	Types	USMDZ1_KTOKS	CHAR	4	0	G/L Account Group
ACCSNEWACC	Types	USMDZ1_ACCOUNTN	CHAR	10	0	New Account
ACCPLTYP	Types	GVTYP	CHAR	2	0	P&L statement account type
ACCRESPP	Types	USMDZ1_RESP_PER...	CHAR	20	0	Person Responsible
ACCRESPU	Types	USMDZ1_RESP_USER	CHAR	12	0	User Responsible
ACCTYP	Types	USMDZ1_ACCTYP2	CHAR	1	0	Account Type
COMPACC	Types	USMDZ1_COMPANY	CHAR	6	0	Company
FSIACC	Types	USMDZ1_FSI	CHAR	10	0	Group Account
FSIACCSTIA	Types	USMDZ1_FSISTAT	CHAR	10	0	Statistical Group Account
FUNCAACC	Types	FKBER	CHAR	16	0	Functional Area
TXITLG	Types	USMD_TXITLG	CHAR	60	0	Description (long text)
TXISH	Types	USMD_TXISH	CHAR	20	0	Description (short text)
USMD_ENI_CHNG_AT	Types	USMD_ENIITY_CHA...	DEC	15	0	Changed On
USMD_ENI_CHNG_BY	Types	USMD_ENIITY_CHA...	CHAR	12	0	Changed By
USMD_ENI_CRID_AT	Types	USMD_ENIITY_CRE...	DEC	15	0	Created On
USMD_ENI_CRID_BY	Types	USMD_ENIITY_CRE...	CHAR	12	0	Created By
.INCLUDE	Types	CI_MDG_S_POG_AC...		0	0	

1.2. Search Account

To get the existing account using search criteria, you can use data model method IF_USMD_MODEL~READ_CHAR_VALUE. In the following, you can find an example code for your reference.

```

DATA: lr_model      TYPE REF TO if_usmd_model_ext.
DATA: lt_sel        TYPE usmd_ts_sel.
DATA: ls_sel        TYPE usmd_s_sel.
DATA: lr_data       TYPE REF TO data.
FIELD-SYMBOLS: <lt_data> TYPE SORTED TABLE.
FIELD-SYMBOLS: <ls_data> TYPE any.
CONSTANTS: lc_incl  TYPE ddsign  VALUE 'I'.
CONSTANTS: lc_equal TYPE ddoption VALUE 'EQ'.
CONSTANTS: lc_model TYPE usmd_model VALUE '0G'.
CONSTANTS: lc_fld_account TYPE usmd_fieldname VALUE 'ACCOUNT'.
* Get model instance
CALL METHOD cl_usmd_model_ext=>get_instance
  EXPORTING
    i_usmd_model = lc_model
  IMPORTING
    eo_instance  = lr_model.

CALL METHOD lr_model->create_data_reference
  EXPORTING
    i_fieldname = lc_fld_account
    i_struct    = lr_model->gc_struct_key
    if_table    = abap_true
    i_tabtype   = lr_model->gc_tabtype_sorted
  IMPORTING
    er_data    = lr_data.
ASSIGN lr_data->* TO <lt_data>.
* Convert end user input into selection for read method
* Here is only simple example
*-----
ls_sel-sign      = lc_incl.
ls_sel-option    = lc_equal.
CLEAR lt_sel.
ls_sel-fieldname = 'ACCTYP'.
ls_sel-low       = '2'.
INSERT ls_sel INTO TABLE lt_sel.
*-----
* Read Account by selection criteria
CALL METHOD lr_model->read_char_value
  EXPORTING
    i_fieldname = lc_fld_account
    it_sel      = lt_sel
    i_readmode  = if_usmd_model_ext=>GC_READMODE_ALL_INACT
  IMPORTING
    et_data    = et_data.

```

2. Read Change Request and Approval Information

2.1. Read Change Request by G/L Account Key

In MDG, a change request is the central place to get all information. Therefore, in order to get the approval information, we have to read a change request first. Method

IF_USMD_CREQUEST_API~RETRIEVE_CREQUEST helps you to get this information. In the following, you can find an example codehow to get the information

```

DATA:
    lt_entity          TYPE usmd_ts_value,
    ls_entity_value   TYPE usmd_s_value,
    lt_crequest       TYPE usmd_ts_crequest.

CONSTANTS:
    lc_model          TYPE usmd_model   VALUE 'OG',
    lc_fld_coa        TYPE usmd_fieldname VALUE 'COA',
    lc_fld_account    TYPE usmd_entity  VALUE 'ACCOUNT',
    lc_entity_account TYPE usmd_entity  VALUE 'ACCOUNT'.

* Set Key
ls_entity_value-fieldname = lc_fld_coa.
ls_entity_value-value = 'INT'.
INSERT ls_entity_value INTO TABLE lt_entity.
ls_entity_value-fieldname = lc_fld_account.
ls_entity_value-value = '0000113100'.
INSERT ls_entity_value INTO TABLE lt_entity.

* Read Change request by object
cl_usmd_crequest_api=>if_usmd_crequest_api~retrieve_crequest(
    EXPORTING
        iv_model      = lc_model
        iv_entity      = lc_entity_account
        it_entity_key = lt_entity
        if_activeonly = abap_false
    IMPORTING
        et_crequest   = lt_crequest ).

```

2.2. Read Workflow Log by Change Request

Finally, you are ready to read the approver(s) and approval user decision for G/L Account.

The following methods can be used here:

Get the workflow item of the change request:

```
CL_USMD4_CREQUEST_PROTOCOL =>WORKITEMS_TO_OBJECT
```

Get the work item details:

```
CL_USMD4_CREQUEST_PROTOCOL =>GET_WORKITEM_DETAIL
```

In the following, you can find a sample code how to get this information:

```

CONSTANTS: lc_bor_bus2250 TYPE swo_objtyp VALUE 'BUS2250',
           lc_index      TYPE swc_index  VALUE '000000',
           BEGIN OF lcs_witype,
             w TYPE sww_witype VALUE 'W', "Dialog task
             b TYPE sww_witype VALUE 'B', "Background tasks
             f TYPE sww_witype VALUE 'F', "Subworkflow
           END OF lcs_witype.
FIELD-SYMBOLS: <ls_items>      TYPE swr_wihdr.

DATA: lt_items                TYPE usmd4_t_swr_wihdr,
      lt_message_struct      TYPE usmd4_t_swr_mstruc,
      lt_simple_container    TYPE usmd4_t_swr_cont.
      ls_wf_result           TYPE usmd4_s_alv_wf_result,
      ls_message_struct      TYPE swr_mstruc,
      ls_workitem_detail     TYPE swr_widtl,
      l_action               TYPE usmd_crequest_action,
      ls_simple_container    TYPE swr_cont.
      l_objkey               TYPE swo_typeid,
      l_return_code          TYPE sysubrc.

* Build object key of BOR object
CONCATENATE iv cr number lc index INTO l_objkey.
* Get list of workflow(s) to current change request
CALL METHOD cl_usmd4_crequest_protocol=>workitems_to_object
EXPORTING
  i_objtype      = lc_bor_bus2250
  i_objkey       = l_objkey
  i_top_level_items = '1'
IMPORTING
  e_return_code  = l_return_code
CHANGING
  t_worklist     = lt_items.
CHECK l_return_code EQ 0.
* Get only the dialog work item
LOOP AT lt_items ASSIGNING <ls_items> WHERE wi type = lcs_witype-w.
*** Approver info stores in WI_AAGENT(user name) and WI_AA_NAME(user description)
* To get the approval user decision, need to read workflow item details
CLEAR: ls_wf_result, ls_workitem_detail.
CALL METHOD cl_usmd4_crequest_protocol=>get_workitem_detail
EXPORTING
  i_workitem_id   = <ls_items>-wi_id
IMPORTING
  e_workitem_detail = ls_workitem_detail
  e_return_code     = l_return_code
CHANGING
  t_message_struct = lt_message_struct.
* Check result
CHECK l_return_code EQ 0.
* Get detail information of current workitem for user decisions
CLEAR: l_return_code, lt_message_struct, lt_simple_container.
CALL METHOD cl_usmd4_crequest_protocol=>read_container

```

```

EXPORTING
  i_workitem_id = <ls_items>-wi_id
IMPORTING
  e_return_code = l_return_code
CHANGING
  t_simple_container = lt_simple_container.
* Check result
  CHECK l_return_code EQ 0.
* Read the result of a user decision if available
  READ TABLE lt_simple_container INTO ls_simple_container
    WITH KEY element = 'ACTION_RESULT'.
  IF sy-subrc EQ 0 AND
    NOT ls_simple_container-value IS INITIAL.
    l_action = ls_simple_container-value.
*-----
*@@@ Description of User decision could be read from table USMD220T and
* it could be stored in ls_wf_result-ddtext
*-----
  ENDIF.
* Workitem information to result list
  ls_wf_result-ws_id = <ls_items>-wi_id.
  MOVE-CORRESPONDING ls_workitem_detail TO ls_wf_result.
  MOVE-CORRESPONDING <ls_items> TO ls_wf_result.
  APPEND ls_wf_result TO et_wf_log.
ENDLOOP.

```

Table USMD220C stores description of action results (data element: USMD_CREQUEST_ACTION)

Display View "Edit Actions": Overview			
Edit Actions			
Action	Description	Pushbutton Text	Quick Info Text
01	Agree	Agree	Agree
02	Disagree	Disagree	Disagree
03	Approve	Approve	Approve
04	Reject	Reject	Reject
05	Finalize Processing	Finalize Processing	Finalize Processing
06	Send for Revision	Send for Revision	Send for Revision
07	Resubmit	Resubmit	Resubmit
08	Withdraw	Withdraw	Withdraw
09	Activate	Activate	Activate
10	Send for Revision	Send for Revision	Send for Revision
21	Successfully executed	N/A	N/A
22	Successfully executed with warning	N/A	N/A
23	Failed	N/A	N/A
31	Activation successful	N/A	N/A
32	Activation failed	N/A	N/A
33	Activation failed for snapshot	N/A	N/A
80	MDG Example: Check Landing Quota	Check Quota	Check Landing Quota by Specialist

Additional Information

Links

[FPM on SCN](#)

[MDG Guides on Service Market Place](#)

How-to Guides

[Extensibility Options for SAP Master Data Governance](#) → [Financial Data](#)

SAP Notes

- [1637249](#) specifying required information for OSS support
- [2105467](#) specifying required information for Performance Issues

Version History

- 1.1 – Update new SAP Community links
- 1.0 – First release of the document

Copyright

© Copyright 2014-2016 SAP SE. 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.