FLM Authorizations and Security

Table of Contents

- Security and Authentication Methods
  - Single Sign On (SSO)
  - URL Encryption
  - Mapped Users in Enterprise Portal
- Authorizations
  - Overview
  - FLM Form Authorization
  - Limiting Access to Specific FLM Tables.
  - Minimum Authorization for FLM Portal Access
  - Create a read only role for FLM remote access support (FLM_READONLY)
- User Groups
  - Defining User Groups
  - Linking Users to User Groups
- User Exits
  - Authorization Customer Level User Exit

Security and Authentication Methods

FLM components use the underlying SAP NW mechanisms for user authentication and encryption of communication channels (i.e. SSL/HTTPS).

Single Sign On (SSO)

If setting up SSO on a Dual Stack NW700, the attached PDF will provides some guidance . There are a couple of comments in the PDF that should be taken note of.

SSO Tickets In a Dual Stack (COMMENTED VERSION).pdf

Once this is completed, the JCO Data Connection can be configured to use Ticket Logon,

Please note that the MetaData Logon still requires a Username/Password logon.

This will now mean that not only will the Logged on user credentials be passed in the JCO Data, but Remote Debugging can also happen for user that signed on and using the JCO Connection.

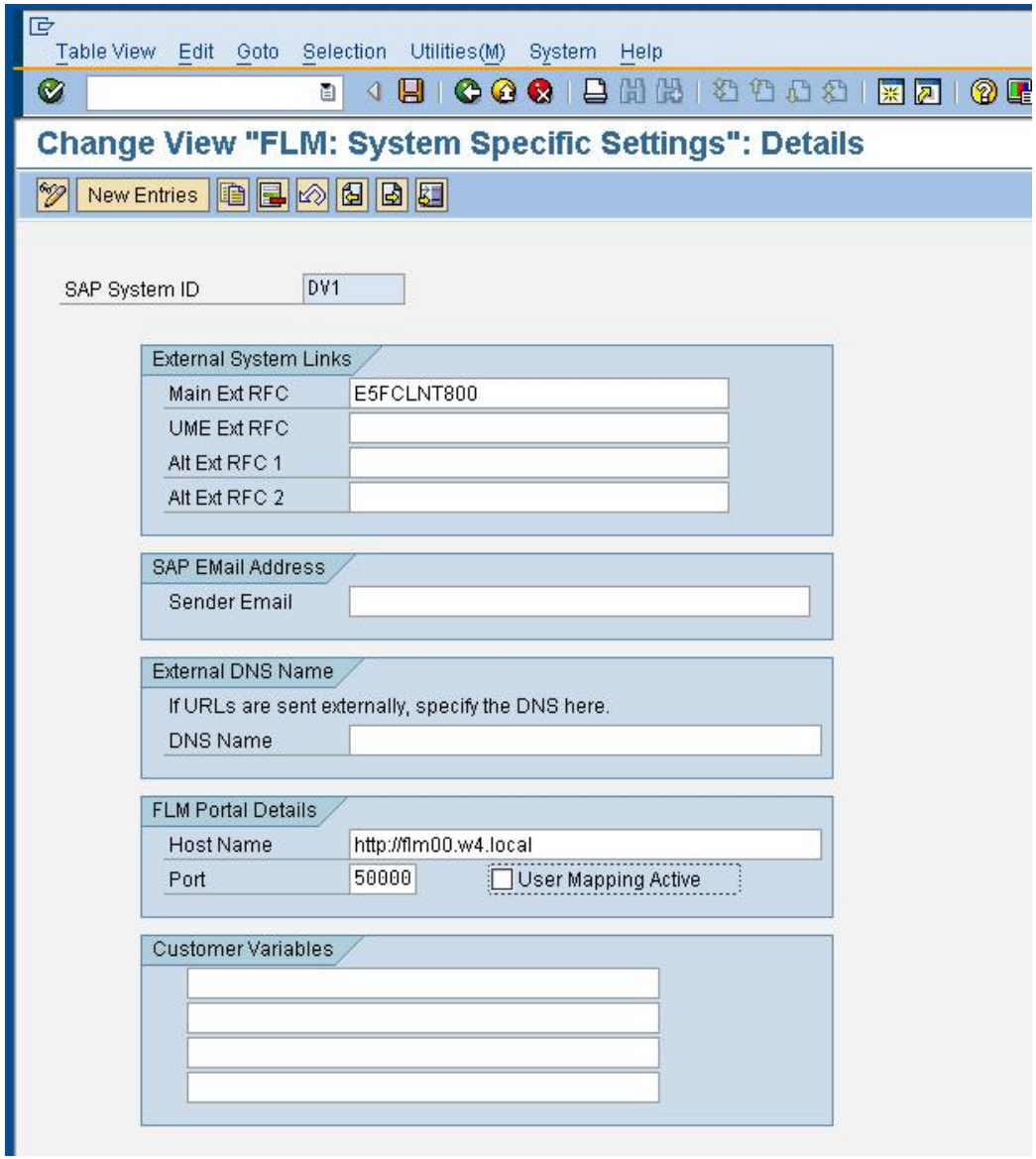
URL Encryption

If, in an online scenario, a user for example fails to approve a form within a given time frame, a reminder email will be sent out containing a URL link to the form. For security reasons, this URL is encrypted according to the encryption key entered with each customer code. The encryption key can be any 14-letter combination that does not include the same letter twice and can be found in the FLM IMG Customer Code settings.

Mapped Users in Enterprise Portal

If users are using Enterprise Portal their user ids must either exist in the ABAP back end and have the correct FLM authorization or the Portal DB (or LDAP) user must be mapped to a user that exists in the ABAP backend that has the correct FLM authorizations.

If the user **does not exist** in the ABAP backend make sure you fill in the FLM Portal Details in the 'FLM: System Specific Settings' IMG activty and check 'User Mapping Active'. You should create an entry for each system in your Landscape.



Authorizations

Overview

Regardless of how users access forms in FLM they need an authorization profile carrying a set of form categories, form types and activities.

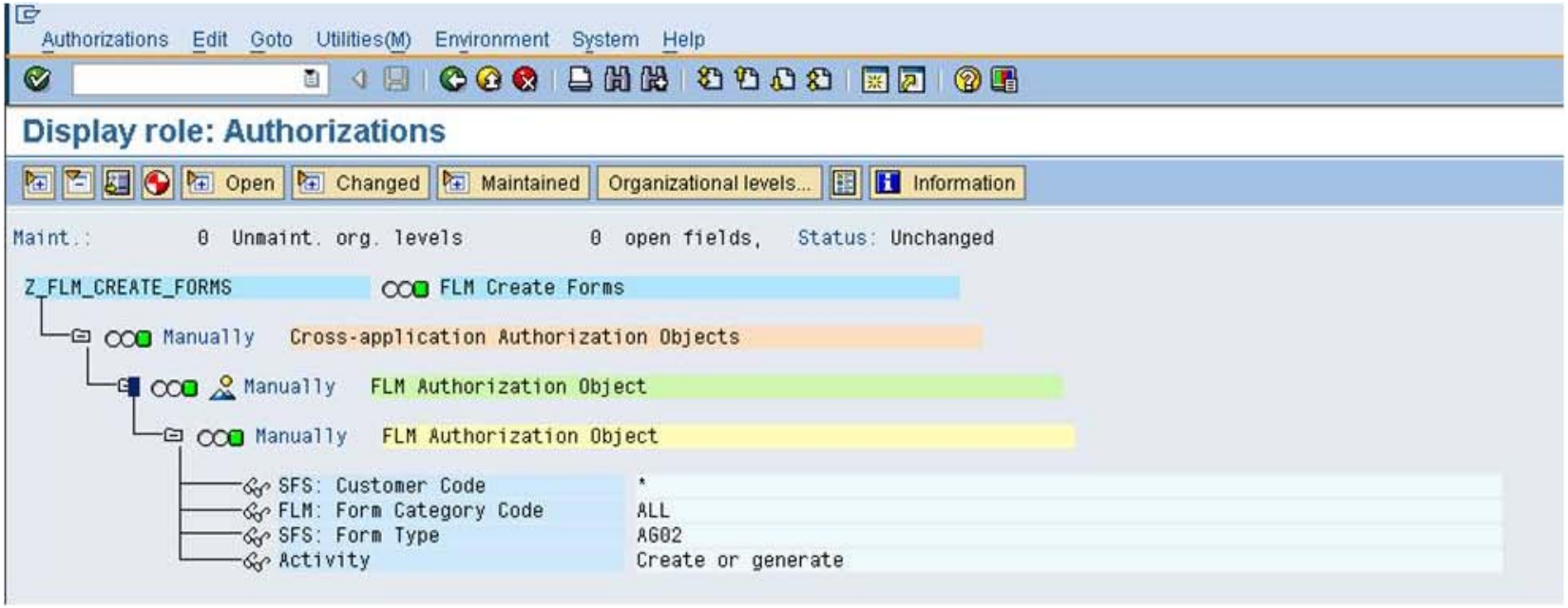
You can find step by step instructions on setting up the Authorization Object and Admin, Developer and User Roles in Section 6 of the Installation Guide.

Commonly Used Roles:

- **FLM Admin** - This role needs the z/flm/0001 object and S_TCODE with /FLM/*.
- **FLM Developer** - The same as above as well as a list of transactions needed for development which can be found in the installation guide.
- **FLM User** - The role needs at least the z/flm/0001 authorization object.
- **FLM Offline** - This role needs the z/flm/0001 authorization object and is used by the Offline User (inbound email processing).
- **FLM Portal** - The user defined in the Java Connector needs at the S_RFC object. There is more detail below.
- **FLM ReadOnly** - This role contains the z/flm/0001 object with Activity set to 'Display' to give access to remote support users.

FLM Form Authorization

This is based on the authorization object defined in customizing under SPRO/Cross-Application Components/General Application Functions/FLM/Initialize Customer Code/Set Customer Code. In a standard installation this is Z/FLM/0001 defined in SU21.



In the example above, you would grant form create access to form category 'ALL' and form type 'AG02'. You can mix and match form categories and form types freely to achieve the effect you require.

Activities that you can choose in FLM are 'Create or Generate', 'Change', 'Display', 'Post', 'Archive', 'Reload' and 'Administrator'. These are permitted activities options ‘01’, ‘02’, ‘03’, ‘10’, '24', '25' and '70'. respectively.

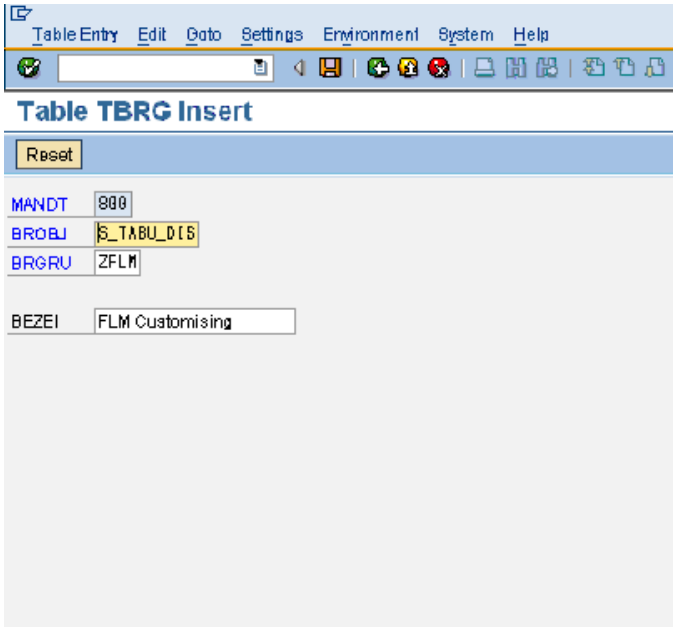
- 'Create' does not imply 'Change'
- 'Change' does not imply 'Display' authorization.
- 'Post' refers to the activity of transferring data from the form into an SAP application using the Form Posting Engine.
- 'Archive' and 'Reload' are used by the FLM Archiving Tools.
- 'Administrator' is required to change owner or status of a form in the dashboard.

Use transaction PFCG to create these roles and assign them to users. Don't forget to do the user comparison step to complete the assignment. This short video illustrates the process of setting up authorizations: FLM Create Authorization.swf

Limiting Access to Specific FLM Tables.

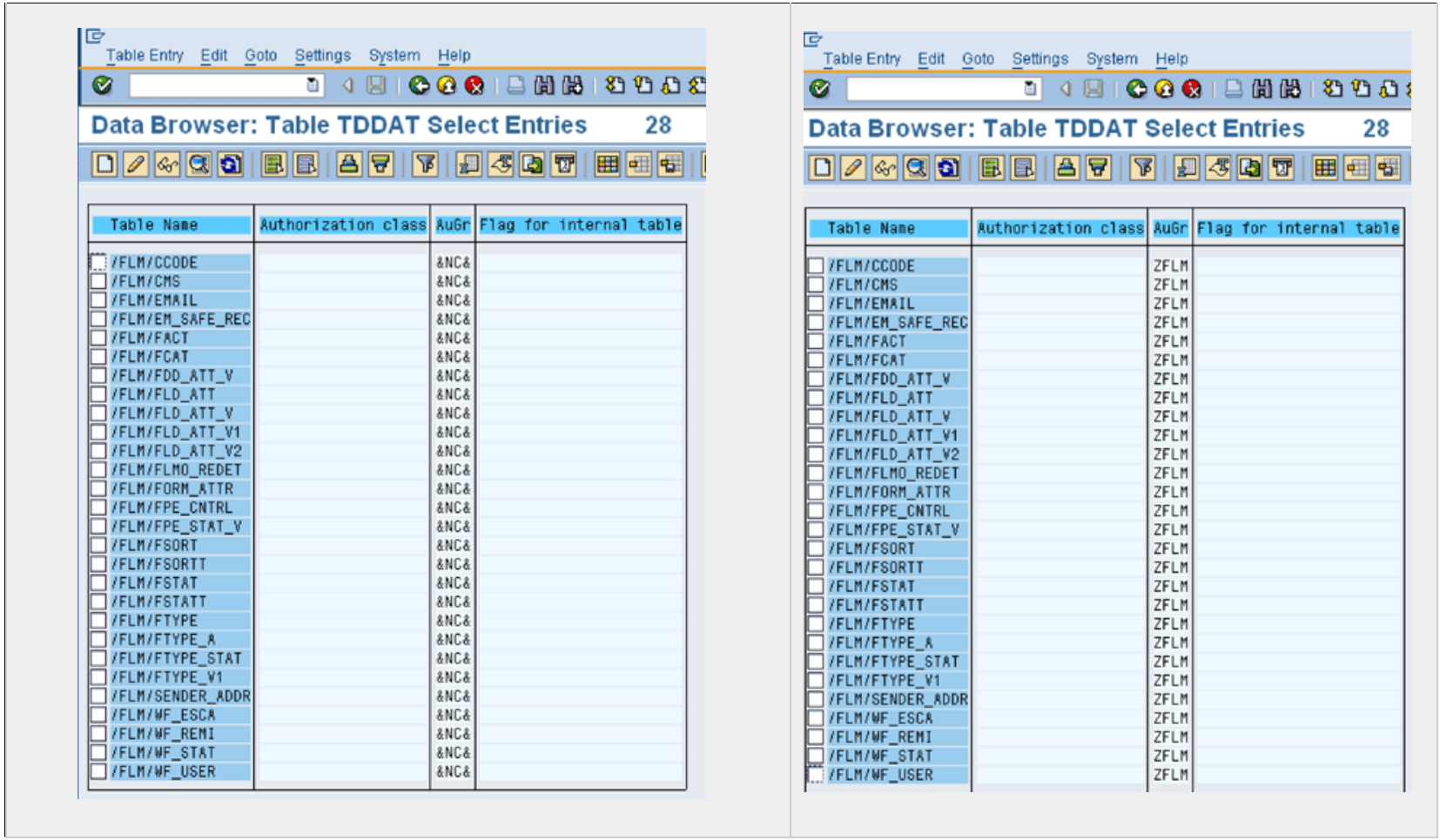
If you would like to limit a user to a specific set of FLM tables you can do this by defining an Authorization Group, we use ZFLM as an example below.

In [SE16] Insert entry in table TBRG as below to define a class for FLM customising tables

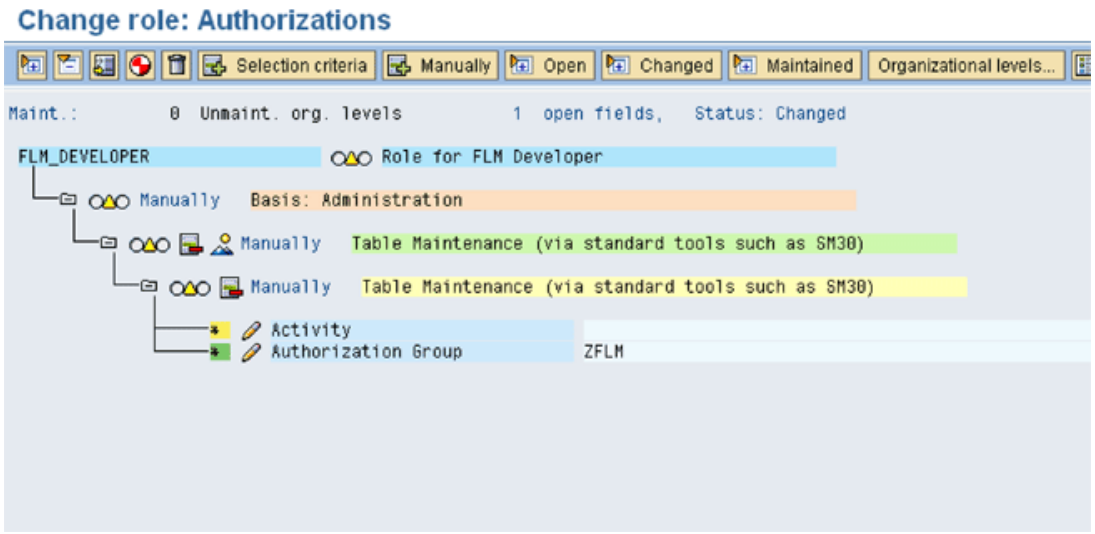


In [SE16] Change entries in table TDDAT for each /FLM/* table – change the authorization group from &NC& to ZFLM

Before	After



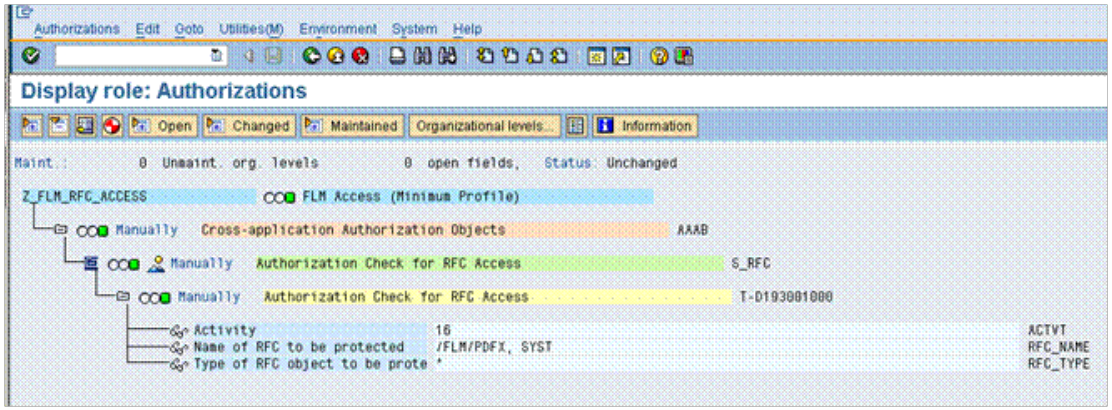
In [PFCG] Add the authorization object S_TABU_DIS with activity '*' and Authorization Group 'ZFLM' to an the desired role as normal



Minimum Authorization for FLM Portal Access

In addition, users accessing forms via the FLM Portal need a minimum RFC authorization. Each of these is described in detail below:

This role is based on the authorization object S_RFC



The Name of the RFC to be protected needs to contain SYST and /FLM/PDFX.

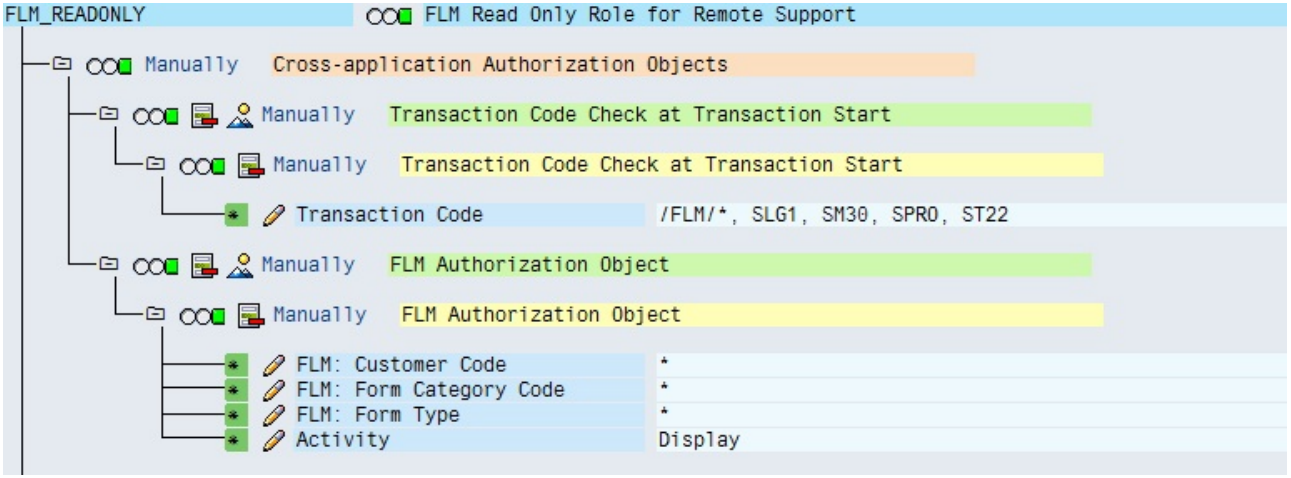
Step-by-step instructions for setting this up are given here.

Create a read only role for FLM remote access support (FLM_READONLY)

In PFCG create a new role FLM_READONLY.

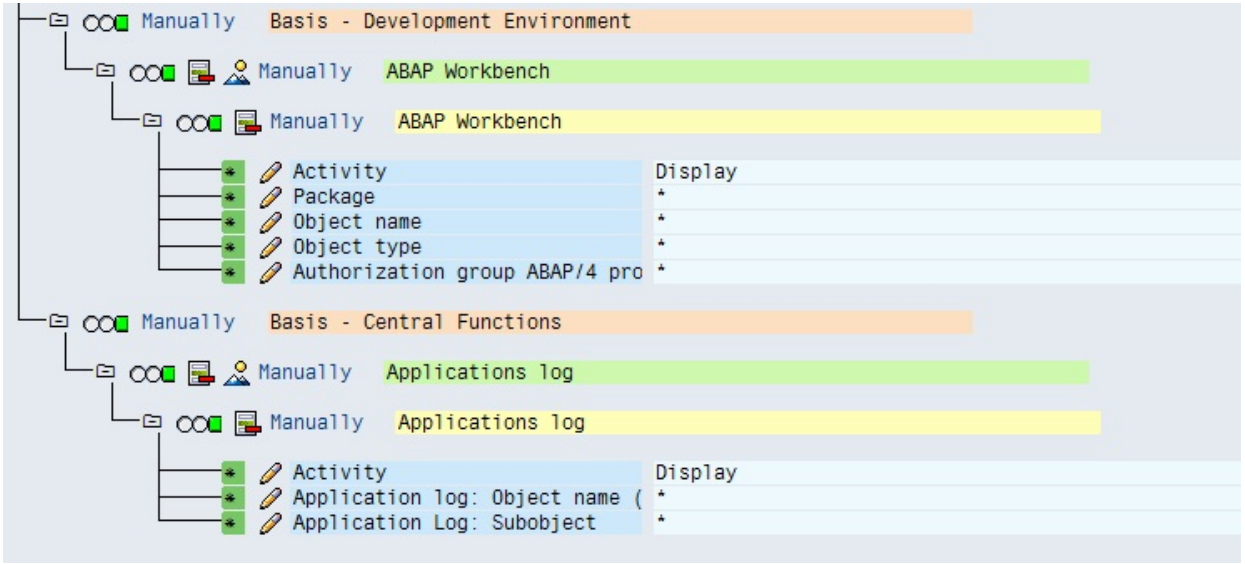
Add the z/flm/0001 object with * for all and Display for Activity.

Add the S_TCODE with transactions /FLM/*, SLG1, SM30, SPRO, ST22.



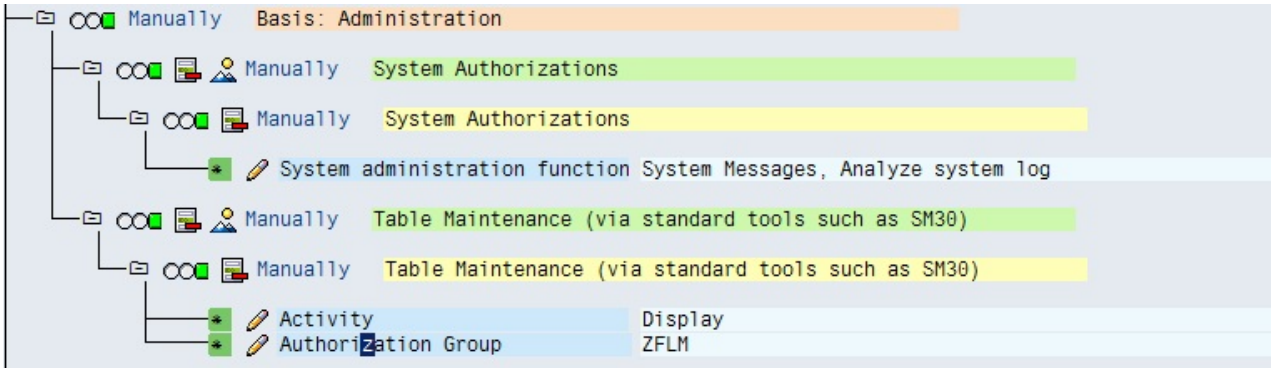
Add S_APPL_Log with * for all and Dispaly for Activity.

Add S_DEVELOP with * for all and Display for Activity.



Add S_ADMI_FCD with SM01 and SM02.

Follow the directions above for **Limiting Access to Specific FLM Tables**. Then add the S_TABU_DIS authorization with Activity set to Dispaly and ZFLM for the group.



User Groups

Defining User Groups

Within FLM there is no configuration table in which to define groups. This means that standard SAP objects like 'position', 'vendor' or indeed 'user group' can be used without the need to maintain that object in an FLM configuration table.

The only constraint in FLM is that the data used to represent the group does not exceed 12 characters, since the data element used to store the form owner has a domain UNAME.

In some cases the new group form owner will be set within the routing logic based on the form status and in other cases the new group could be dynamically derived from data within the form.

User groups are **not used for authorization checking** within FLM, since they are used for routing forms not for triggering new forms processes.

Groups are maintained in transaction: SUGR

Groups are assigned in transaction: SU01

See Group Ownership of Forms in the developers guide for more information.

Linking Users to User Groups

When a user logs on to the FLM Portal and selects the tab to view their Inbox, then by default they see all the

forms that are of a live status and are currently escalated to them: ie. Where the current form owner is equal to their SAP User ID.

There is an FLM user-exit that enables the Inbox to also include forms escalated to relevant user groups. Within this user-exityoue must define how you can select the user groups from the currently logged on user, in order to also include all forms escalated to those groups. [A user may have access to forms in a number of groups.]

FLM uses a standard SAP mechanism to link the user id to the group.

> For example, in an HR scenario, the group may be equal to a 'position' or 'organisation unit' and you can call standard SAP functions to determine what position or organisational unit the current user belongs to.

In other scenarios you may need to store a link between the user and the user group. This typically involves maintaining a custom table, or using existing functionality on the SAP user master, such as user parameters, roles or user groups. You are able to easily read such user details using BAPI_USER_GET_DETAIL. There are pros and cons of using these 3 approaches:

1. **User parameters.** If you use user parameters then the user can maintain their own parameters if they have SAPGUI access. (This may be desirable or not depending on the business process.)
2. **Roles.** If you use roles this can be useful since businesses will have an existing internal process for maintaining roles. However, the business may have naming conventions around roles that mean that they cannot be used for the user group as they are longer than 12 characters. In this case you either:
    - Need to create a custom authorization around 'user group' and assign it to the role. (This means you need to perform multiple authorization checks when viewing the FLM Inbox, and could affect system performance, or
    - Need to create and maintain a custom mapping table between the role and the user group, which you would not expect to change very often.
3. **User groups.** If you use user groups then the user cannot maintain them themselves and there is no need for any custom mapping table.

The best fit for FLM is usually to use the standard SAP user groups, since the domain for the SAP field has type CHAR12 like UNAME.

User Exits

Authorization Customer Level User Exit

If your user master records exist on a different system, you must take advantage of the FLM Customer Level User Exit called 'Authorization'.

In here you must remotely call a function on the target system where the users exist (as does the FLM authorization object) that performs the similar functionality as the /flm/core=>check_flm_authorization does

on the local system. For example, this code:

```
 *
  data: l_auth_object type  xuobject,
        l_form_cat    type  /flm/fcat_code,
        l_user        type /flm/uname,
        l_ref_fcat    type /flm/fcat_code,
        l_cust_class  type string,
        l_routine     type string.
*---------------------------------------------------------------------*
* Authorisation actually has to pass positively to work:
*---------------------------------------------------------------------*
  ex_result = 4.
  check im_user is not initial.
*---------------------------------------------------------------------*
* Get authorisation object.
*---------------------------------------------------------------------*
  l_auth_object = 'Z/FLM/0001'.
*---------------------------------------------------------------------*
* Some java stacks do not Capitalise the username,
* otherwise will get rc=40 in this case:
*---------------------------------------------------------------------*
  l_user = im_user.
  translate l_user to upper case.
*---------------------------------------------------------------------*
* Execute the check depending on if the form type is supplied:
*---------------------------------------------------------------------*
  if im_ftype is initial.
*
    authority-check object l_auth_object for user l_user
        id '/FLM/CUST'  field im_ccode
        id '/FLM/FCAT'  field l_form_cat
        id 'ACTVT'      field im_activity.
*
  else.
*
    authority-check object l_auth_object for user l_user
        id '/FLM/CUST'  field im_ccode
```

```
          id '/FLM/FTYPE' field im_ftype
          id '/FLM/FCAT'  field l_form_cat
          id 'ACTVT'      field im_activity.
*
  endif.
*---------------------------------------------------------------------*
  ex_result = sy-subrc.
```