Teamcenter Enterprise: Rule System

Volume ITI-MTI-E
Written by
Erich Brauchli

March 2, 2008
Version 2.10

Reflecting Teamcenter® Enterprise 2005

Email: erich@brauchli.info
Internet: http://www.brauchli.info
(Total 21 pages)

Content

1 General Remark 3
2 Message Access Rules 4
2.1 Basic schema 4
2.1.1 Participant 5
2.1.1.1 User 5
2.1.1.2 User Group 5
2.1.1.3 Role 5
2.1.1.4 Dynamic User 6
2.1.1.5 Team 6
2.1.1.6 Team Role 6
2.1.1.7 Team Project 7
2.1.1.8 Team Role Assignment 7
2.1.2 Action (message) 7
2.1.3 Object (class) 7
2.1.4 Condition 7
2.2 General remark and sample on message access rules 9
2.3 Establish your custom Rule System 9
2.3.1 Step 1: Define Functional Groups 9
2.3.2 Step 2: Identify a group hierarchy relative to rules 9
2.3.3 Step 3: Define the Roles relative to Projects 10
2.3.4 Step 4: Describe as narrative the access rights for each functional group 10
2.4 Prepare your Rule System 10
2.4.1 Establish required Message Groups 11
2.4.2 Translate narrative Definition into single Rules 12
2.5 Implement your Rule System 14
3 Location Selection Rules 16
3.1 Physical aspects 16
3.2 Application aspects 16
3.3 The “Location Selection Rule” 17
3.3.1 Arguments 17
3.3.2 Samples 18
3.4 The Vault Administrators Work 19
4 Notification Rules 20
4.1.1 Arguments 20
4.1.2 Samples 21

Entire List of Teamcenter Documents:

ITI-MTI-A	Guidelines for Planning and Operation
ITI-MTI-B	Guidelines for Setup and Configuration
ITI-MTI-C	Guidelines for Oracle Setup and DB Configuration
ITI-MTI-D	Guidelines for Customizing
ITI-MTI-E	Rule System

1 General Remark

This document explains the Teamcenter Enterprise Rule System. It includes recommendations how to set up and maintain the rules.

Rules govern the functional permission for users (Message Access Rules), the use of Vault Locations (Location Selection Rules) and the automatic Notifications to users (Notification Rules); the latter include automatic mails from Teamcenter Enterprise to users.

Rule system relevant objects are:

- Users, Dynamic Users
- User Groups, Teams
- Roles, Team Roles
- Role Assignments, Team Role Assignments
- Hosts
- Host Groups
- Conditions[1]
- Messages
- Message Groups
- Projects

Message Access Rules

Message Access Rules permit an action to a Participant (user) on a certain object under a condition.

Thus the basic system does not allow any action to anybody. There are only rules, which **allow**, but none that deny an action. With other words each rule is a hole in the protection shield; and no other rule can close this hole again.

Location Selection Rules

Location Selection Rules allow the CheckIn or move to Vault Locations, depending on user, host file type and conditions. A priority defines the preferred or default behavior if more than of Location is open for a Vault.

Notification Selection Rules

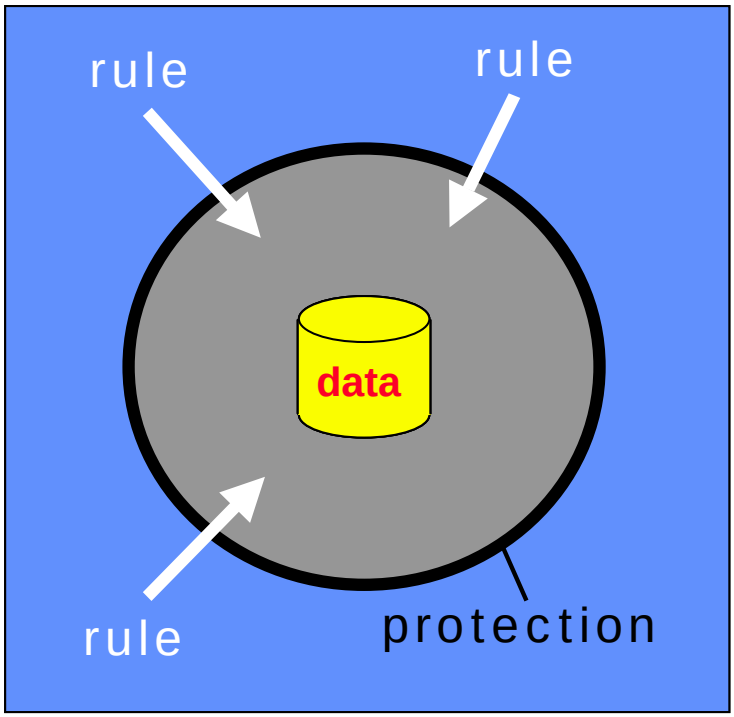
Notification Rules define who will be notified (get an email) on which action on which object and under which condition.

Typical sample for such a rule: If some body deletes an object, which is not his own (and for which he has the right), the owner is notified about that fact.

[1]) Conditions are used in Processes and in Code Control as well.

2 Message Access Rules

2.1 Basic schema



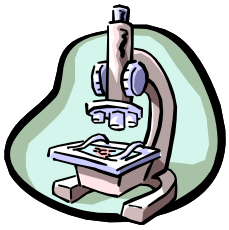
- Data is completely protected by a protection shield ("rules engine").
- Each rule is an exception, a hole in this shield.
- Rules can only allow things, but never deny.
- Denied is what ever is not permitted by a rule.
- Rules cannot override hard coded validation prohibition. This is the case for actions, which lead to illegal situations.

A rule defines in a generic way, that:

Some body

↑

Participant



may do an action

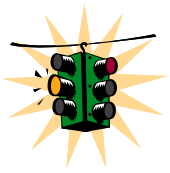
↑

Action
(message)

with a type of objects

↑

Object
(class)



under some condition

↑

Condition

2.1.1 Participant

The participant is a representation of users. This may be either a "*", which means every user, or one of the following: User, User-Group, Team, Role, Team-Role or Dynamic User.

2.1.1.1 User

This is a named individual, which is registered as a Teamcenter Enterprise user. A User is identified by its Teamcenter Enterprise internal name.

A rule with a user as participant is valid only for this individual, which identified himself at login time by the user name and a password.

It is not recommended to create rule for single individuals. This implies a lot of administrative overhead. It is not likely to gain an overview over your rule system, when designed by individuals. And it tends to a high number of rules, which has an impact on performance and keeps administration busy.

2.1.1.2 User Group

A user group is identified by its group name. A rule for a user group is valid for all members of that group.

All rules, which are not "Project" dependent (see lower down), should be created on behalf of user groups, which identify a functional job, like "Process Administrator", "Common User", "User Administrator", …

Whoever has an appropriate function, is then assigned to the relative group. An individual may be assigned to as many functional groups as required.

If the group names reflect the functional requirements, this is an easy way of managing user rights.

It is strongly recommended

- To create functional groups and to assign rules to these groups.
- Assign every body to the group(s) representing his function.
- Do not repeat rules on several groups; thus create basic functional groups and incremental functional groups. This leads to the fact that some function will be represented by a sum of functional groups.
- Assign a person to more than one group if appropriate.

2.1.1.3 Role

By definition "Role" is a role in a project. A typical sample for a role is "Project Leader".

Such a role must be assigned for each singular project to a person or a group of persons. But the same person may be a "Project Leader" in some projects, but not in others.

Special rights for persons in such a project roles have to be defined upon Roles. And the roles will then be assigned to persons or groups for each project. They will be in effect only for objects assigned to the relative project.

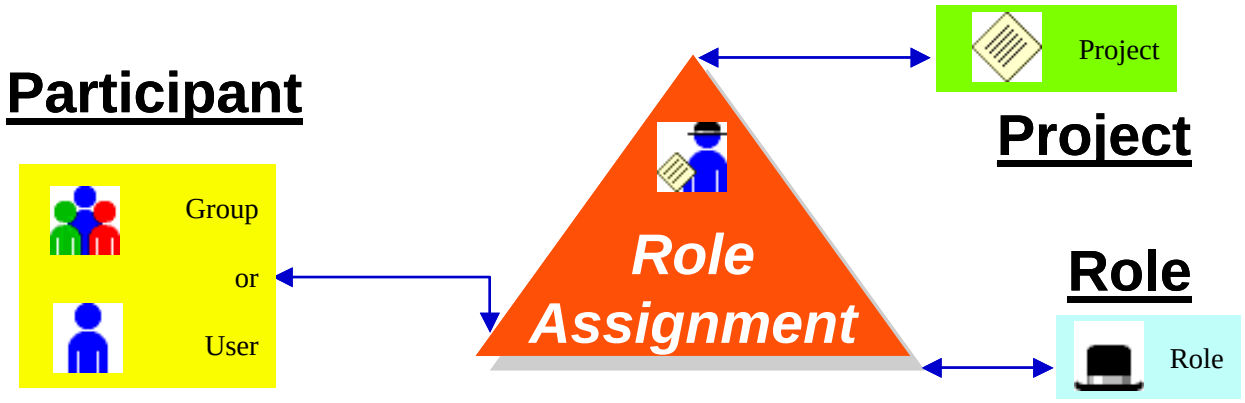
User rights defined on Roles have absolutely no effect for object not assigned to a project nor for objects, where no person is assigned to that specific Role.

It is strongly recommended

- To create Roles and assign rights to these roles.
- Assign person(s) to each role for each project.
- Make the Project-Name a required attribute for Business Items.

But only, if you work in a real project oriented organization, where same functions are assigned to different people for different projects.

NOTE: Role assignments have a participant, while Roles are Participants in Rules.



Role assignments are like relations between **three** objects.

2.1.1.4 Dynamic User

Dynamic Participants are Users, whose names are stored in an attribute of the respective object. Typical samples for such a Participant are “The Creator” or “The Submitter” of a document or a part.

Rules defined on dynamic participants in general give a special right to some one, who did already some thing with the specific object.

Dynamic Participants are defined as the attribute name of objects, which contains the real user name. “$obj.Creator” means the user, whose name is stored in the attribute “Creator” of the object, which is filled by the system at object creation time and then never more changed. “$obj.Submitter” means the user, whose name is stored in the attribute “Submitter” of the object; be aware that the attribute “Submitter” is even a dynamic attribute on Work I 45tkp23654beitter”MCID 53 5ic

2.1.1.7 Team Project

Team Roles are roles just relative to the team.

2.1.1.8 Team Role Assignment

Similar to the standard role assignment, but with much finer granularity, which deals with teams, tema roles and team projects..

Using the Team version of Message Access Rule is only meaningful, if a company really has sch a working organization.

2.1.2 Action (message)


The “Message” field of the rule object defines the activity, the user get the right to execute on the relative objects. This action is finally an object or class “Message”[2].

The field accepts single non-trusted external message names, or message group names. Message groups are collections of both non-trusted external messages and other message groups. This grouping helps to reduce the number of rules and thus increases the overall performance of the system.

A star (“*”) in this field means “all actions”:

2.1.3 Object (class)

The class field in a rule indicates the type of object, upon which the action may be executed. It is defined as a class name in the hierarchical object class tree.

There is an option to define just this class or to include the entire branch under this class. It is not common to use the “class only” feature; the rule system is much smarter using more “class inherit” rules.

A star (“*”) in this field means “all object and relation classes”.

Note: Special concerns on rules upon relation classes and relationships are obvious. To allow the creation of a relationship we need three rights: the access right to “relate” both the Right and the Left class object, and the access right to “create” the relation class object.

2.1.4 Condition

The condition object mentioned in a message access rule object restricts the right to cases, where this condition results in a “TRUE” value.

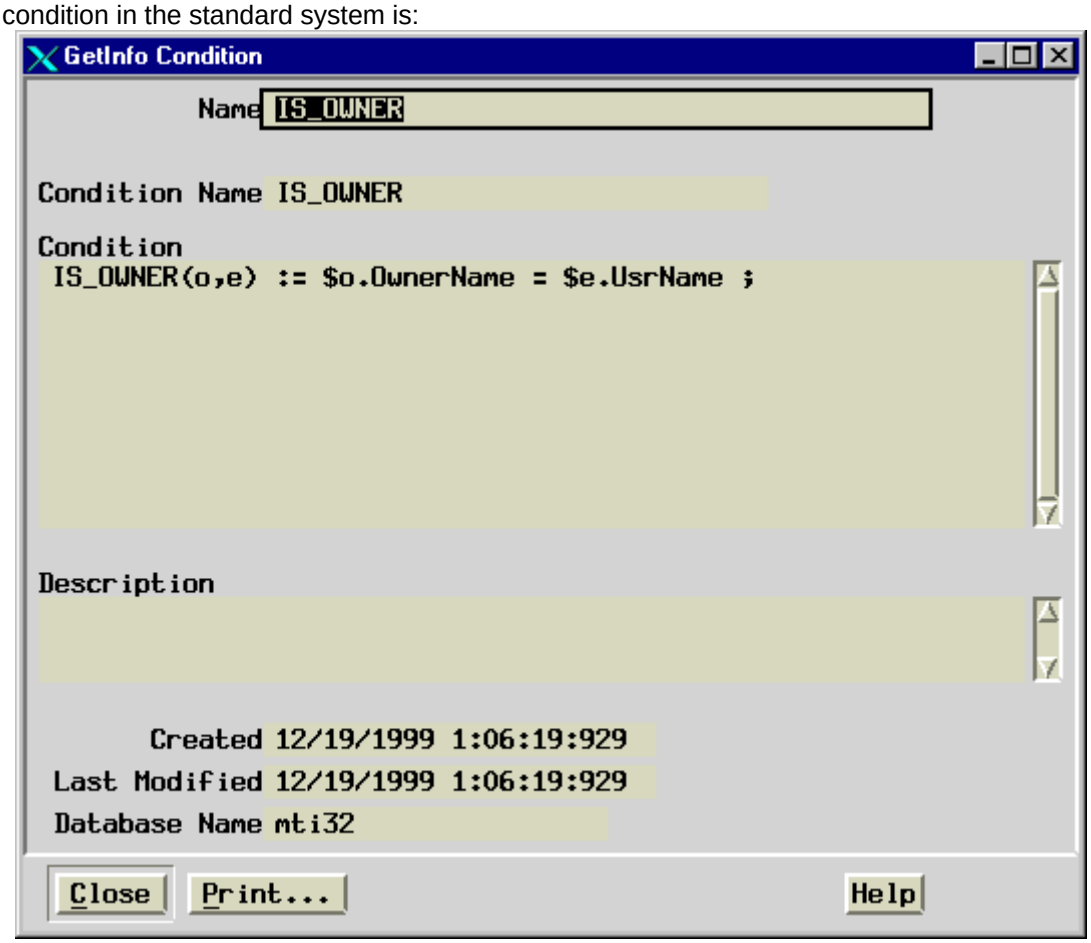
The system accepts in this field the two symbolic conditions “TRUE” and “FALSE”[3] and any defined condition object, which is defined with **two** arguments. The two arguments are the object (usually marked by “obj” or “o”) and the environment (“env” or “e”), which is the session object containing information on the acting user.

---

[2]) Only non-trusted external messages are subject to the rule system and therefore only these messages are allowed as value in this field.

[3]) The “FALSE“ condition does not make any sense in a rule. It is used (if ever) to temporarily turn off a rule without deleting it. A “FALSE” rule increases the number of rule, without any change to the system. Be aware, that a “FALSE” rule will **not** deny any access granted by another rule!

A typical sample condition in the standard system is:



This condition is true, if the acting user owns the object.

Argument "object":

This argument represents the real object, on which the rule is applied, each time when a rule is checked against an object (i.e. each time a user tries to execute an action, where the relative rule could be involved). This permits in the condition to check any of the real objects attribute values.

Argument "environment":

This argument represents the "session object", which mainly represents the real actual user and the actions context. The "session object" contains attributes about the login user and the action context and membership in groups and teams, but no information on roles or projects. Thus even a rule is defined for a user group, this argument is relative to the user, not the group.

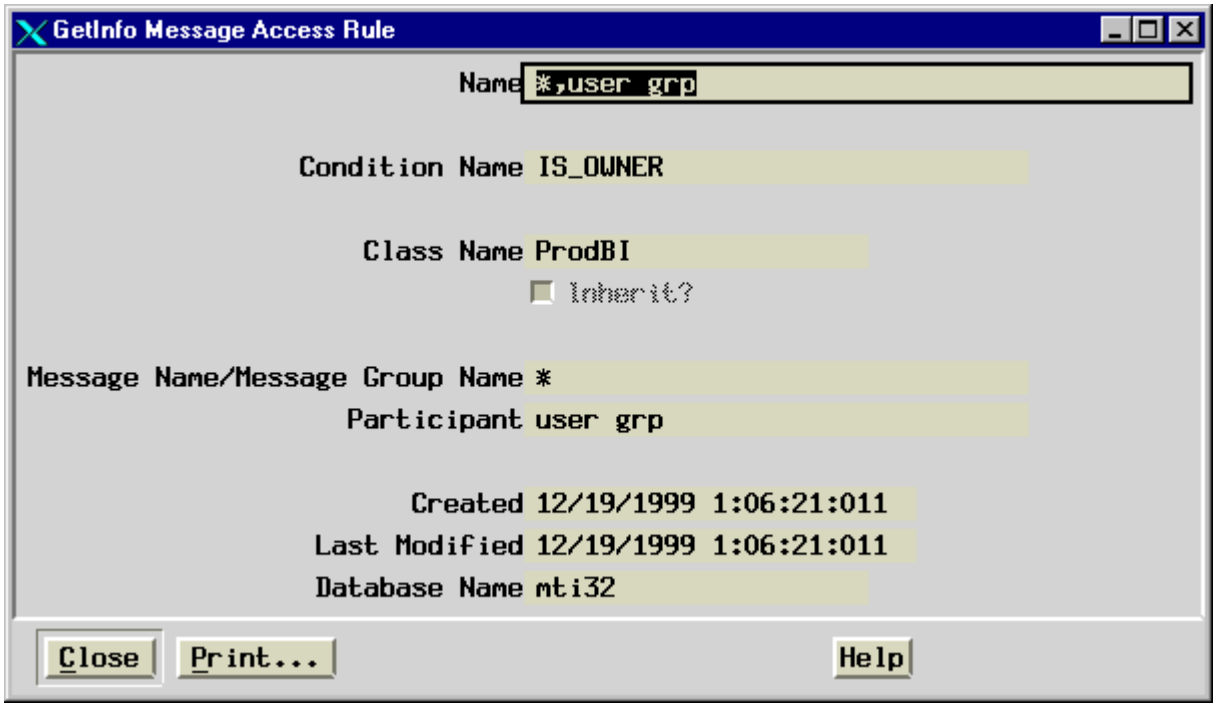
It is recommended to use the Description field on your own custom conditions. For performance reasons the conditions in a running system are kept in the rule cache file, but the description is not passed into that file. Thus it is a pure informational field, helps you to easily identify standard and custom rules and should describe in an understandable form eventually complex condition and/or what they are used for.

2.2 General remark and sample on message access rules

It is strongly recommended

- That you design your rule system with as few rules as possible**. This is performance relevant**.
- To reduce rules, group users and assign users to more than one group where appropriate.
- Group Messages into Message-Groups recursively.

Sample Rule:



This standard rule allows every action for users in the "user grp" group on any "Product Business Item" owned by him.

2.3 Establish your custom Rule System

This is real definition work to be done by technical or security related person in PDM steering committee. The relative people do not need more knowledge about Teamcenter Enterprise than an average user.

2.3.1 Step 1: Define Functional Groups

Define the list of functional user groups in your company and give them significant names according your naming convention[4]. Start with a standard basic user group (this may be the Teamcenter Enterprise standard group "user grp", which is intended by Teamcenter Enterprise for this purpose).

Other functional groups should are added to complete all user profiles necessary in your company.

2.3.2 Step 2: Identify a group hierarchy relative to rules

Identify for each group (except the basic one), which other groups' rights must be included for this group. The goal is to identify a group hierarchy relative to user rights, so that we can later for each group define, which is its relative parent group and which are the additional rules for this group.

[4]) A good convention for User Group names is: Start with an upper case letter, and add a space and „grp".

The goal is to define for each group only some additional rights relative to another (parent) group. This will reduce the number of rules per group and define to which other group all members of this group have to be assigned too.

2.3.3 Step 3: Define the Roles relative to Projects

If you are working in a project-oriented environment, define the roles, which have to be assigned in each project. For each role you may define an underlying generic user group, in which users assuming the role should be too.

If appropriate to your company define team roles the similar way.

2.3.4 Step 4: Describe as narrative the access rights for each functional group

Start again with the basic group and define as narrative text, what users in this group must be able to do. Formulate this in the form of description of real user activity on object type (again narrative, not class names) and where appropriate express a restriction.

Then step by step go through the group hierarchy and do the same for each functional group, but just add the difference relative to the parent group.

Do the same thing for Project Roles; again define the incremental right relative to the underlying group.

It is strongly recommended to add an identifier to each line in this list. This identifier will later be added to the description field in each rule object, so that there is a reference to this narrative definition.

This should result in a set of tables like this:

Group XY		Parent Group: YZ	
Ident	Activity	Type of things	Restriction
G-1.1	do every thing	all, except Suppliers	if they own it
G-1,2	Checkin	Documents, Parts	if they own it
G-1.3	Checkout	Documents, Parts	if …..
G-1.4	Submit	Documents, Parts	if ….
G-1.5	Query and View	every thing	if not flagged confidential

and

Role XY		Underlying Group: YZ	
Ident	Activity	Type of things	Restriction
R-3.1	Revise	Documents	
R-3,2	Assign Responsible Person to	Parts	if in Vault XY

2.4 Prepare your Rule System

This step is to be done by a technically experienced person knowing Teamcenter Enterprise concepts, static and dynamic data model.

2.4.1 Establish required Message Groups

Verify, that any action description in the above tables may be represented as a “Message Group”. If this is not the case, add Message Groups as required[5].

Base your own groups on the Teamcenter Enterprise standard Message Groups as far as possible[6]. Avoid wherever possible to add directly messages into your customized Message Groups, except your own custom messages.

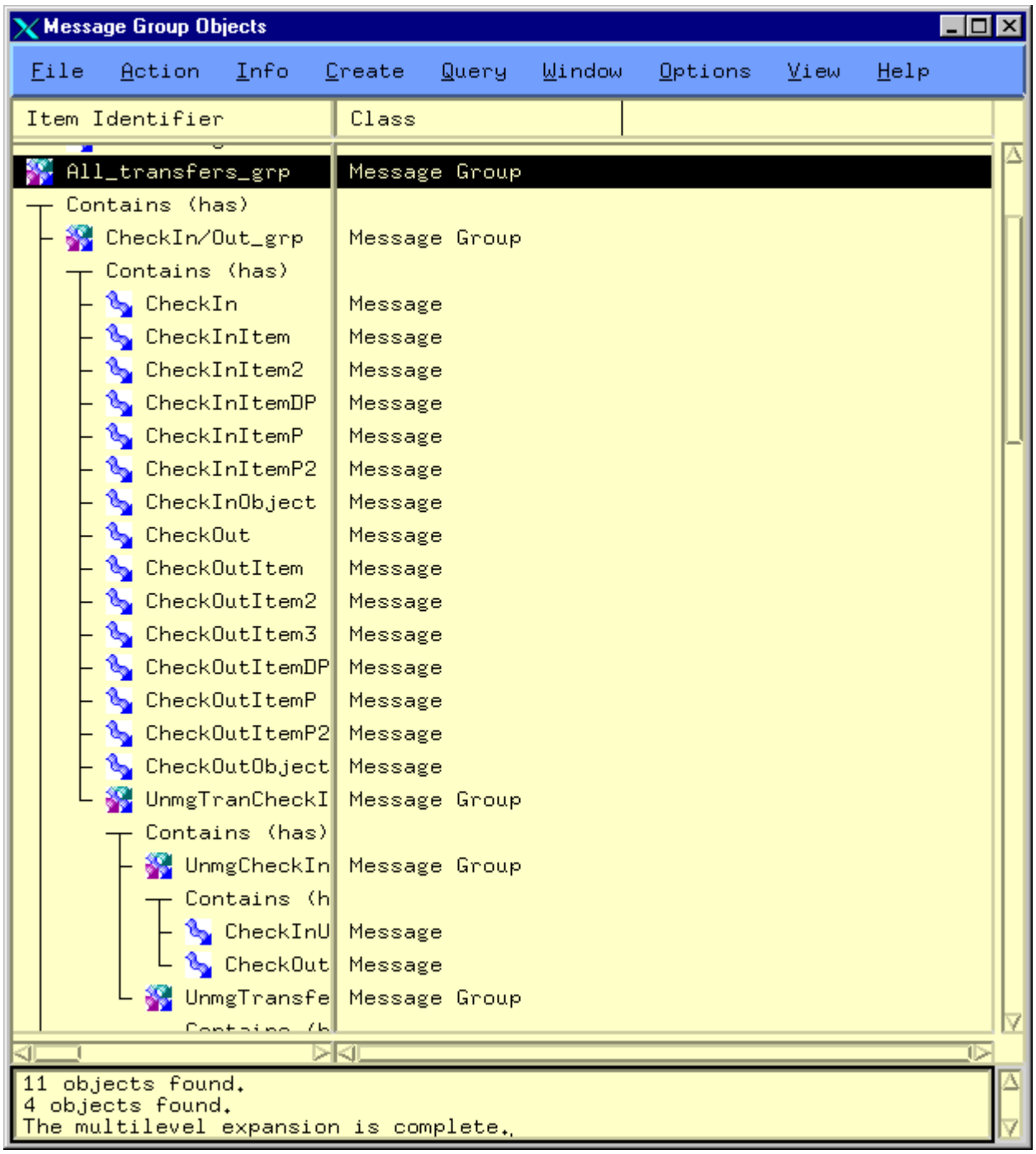
Add a list of additional Message Groups with narrative content to the definition tables above, the create them:

Create, if necessary, new Message Groups through the user interface:

[5]) A good convention for Message Group names is: start with a upper case letter and add after an underscore “grp” or even “msggrp”

[6]) This will make easier your rule maintenance whenever Teamcenter Enterprise adds new messages to their groups in future releases.

Put messages into Message Groups by drag and drop:

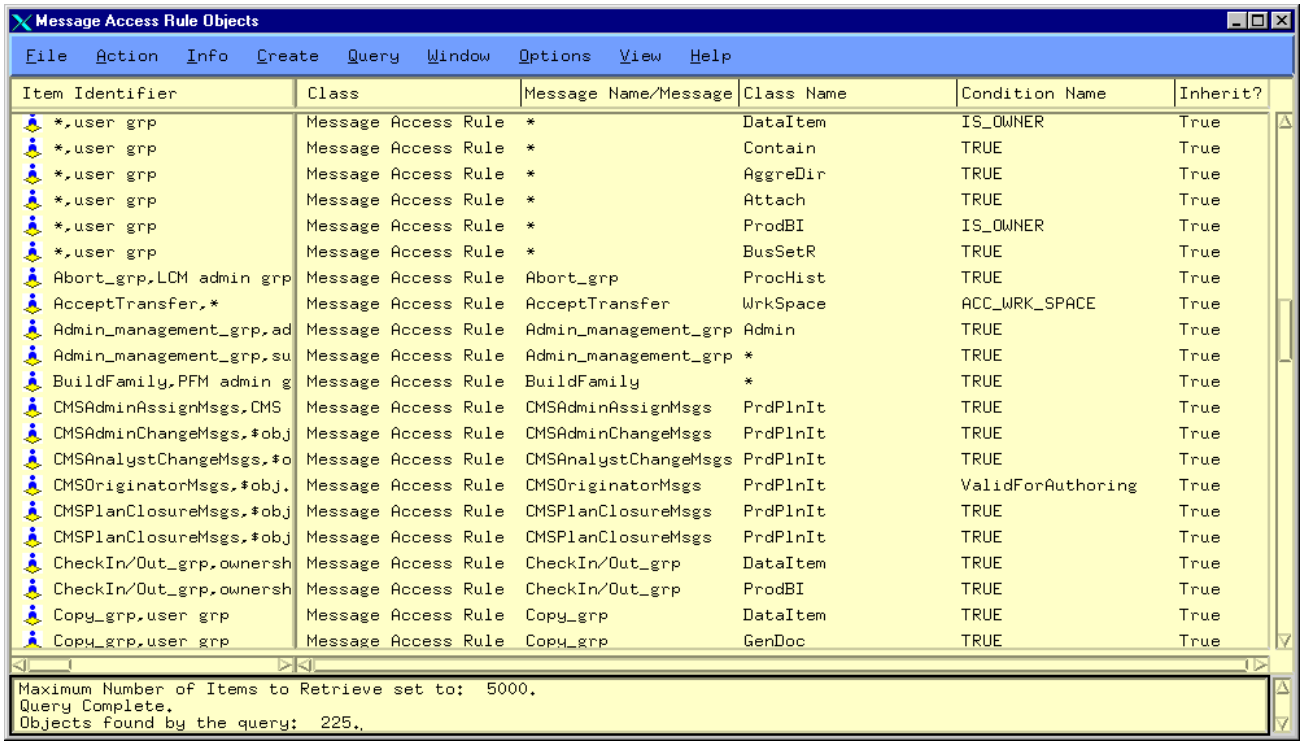


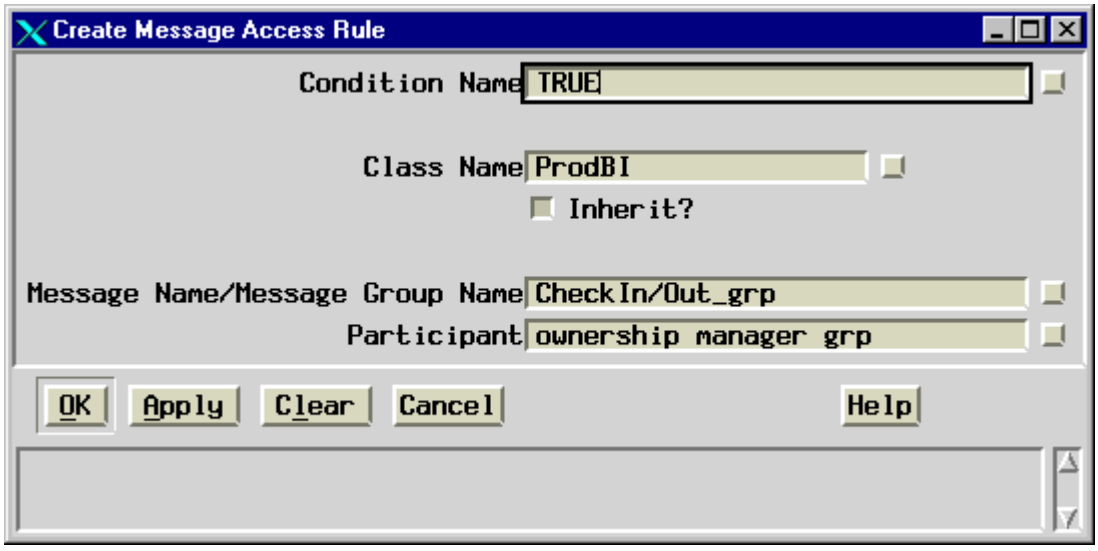
2.4.2 Translate narrative Definition into single Rules

Translation into Teamcenter Enterprise Rules needs some knowledge about Teamcenter Enterprise data model and message concept.

It is important, to generate as few rules as possible to reduce validation time for each menu click. It is allowed to change or delete standard Teamcenter Enterprise rules and to add new ones.

Create the list of rules first on paper, and identify them with the identifiers from the original tables. Then create them in the system:

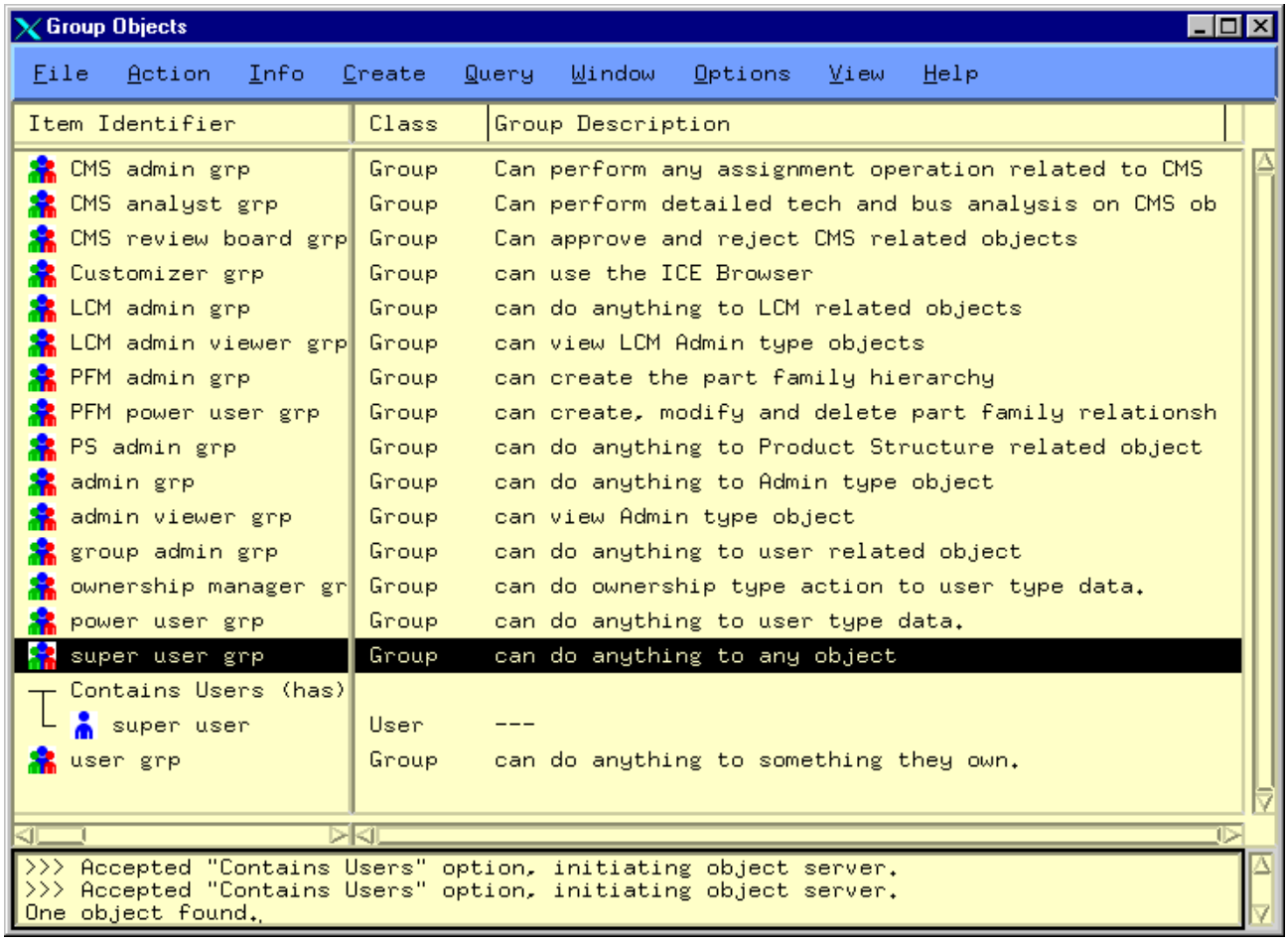

Message Access Rule Objects
File Action Info Create Query Window Options View Help
Item Identifier | Class | Message Name/Message | Class Name | Condition Name | Inherit?
*,user grp | Message Access Rule | * | DataItem | IS_OWNER | True
*,user grp | Message Access Rule | * | Contain | TRUE | True
*,user grp | Message Access Rule | * | AggreDir | TRUE | True
*,user grp | Message Access Rule | * | Attach | TRUE | True
*,user grp | Message Access Rule | * | ProdBI | IS_OWNER | True
*,user grp | Message Access Rule | * | BusSetR | TRUE | True
Abort_grp,LCM admin grp | Message Access Rule | Abort_grp | ProcHist | TRUE | True
AcceptTransfer,* | Message Access Rule | AcceptTransfer | WrkSpace | ACC_WRK_SPACE | True
Admin_management_grp,ad | Message Access Rule | Admin_management_grp | Admin | TRUE | True
Admin_management_grp,su | Message Access Rule | Admin_management_grp | * | TRUE | True
BuildFamily,PFM admin g | Message Access Rule | BuildFamily | * | TRUE | True
CMSAdminAssignMsgs,CMS | Message Access Rule | CMSAdminAssignMsgs | PrdPlnIt | TRUE | True
CMSAdminChangeMsgs,$obj | Message Access Rule | CMSAdminChangeMsgs | PrdPlnIt | TRUE | True
CMSAnalystChangeMsgs,$o | Message Access Rule | CMSAnalystChangeMsgs | PrdPlnIt | TRUE | True
CMSOriginatorMsgs,$obj. | Message Access Rule | CMSOriginatorMsgs | PrdPlnIt | ValidForAuthoring | True
CMSPlanClosureMsgs,$obj | Message Access Rule | CMSPlanClosureMsgs | PrdPlnIt | TRUE | True
CMSPlanClosureMsgs,$obj | Message Access Rule | CMSPlanClosureMsgs | PrdPlnIt | TRUE | True
CheckIn/Out_grp,ownersh | Message Access Rule | CheckIn/Out_grp | DataItem | TRUE | True
CheckIn/Out_grp,ownersh | Message Access Rule | CheckIn/Out_grp | ProdBI | TRUE | True
Copy_grp,user grp | Message Access Rule | Copy_grp | DataItem | TRUE | True
Copy_grp,user grp | Message Access Rule | Copy_grp | GenDoc | TRUE | True
Maximum Number of Items to Retrieve set to: 5000.
Query Complete.
Objects found by the query: 225.

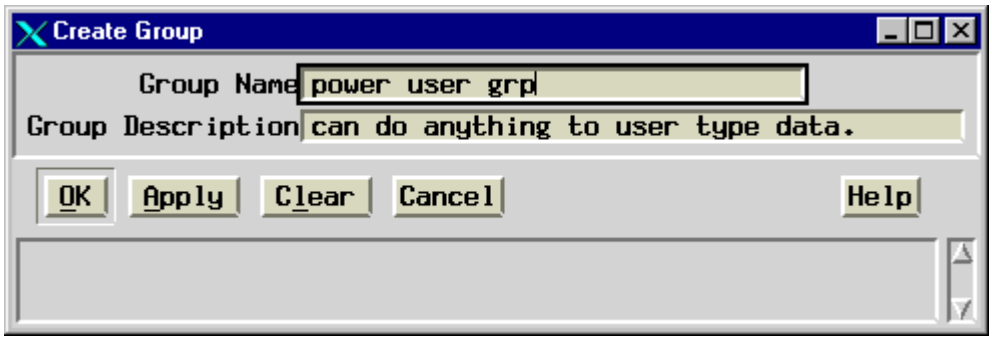

Create Message Access Rule
Condition Name TRUE
Class Name ProdBI
Inherit?
Message Name/Message Group Name CheckIn/Out_grp
Participant ownership manager grp
OK
Apply
Clear
Cancel
Help

2.5 Implement your Rule System

This is normal work of a Teamcenter Enterprise administrator.

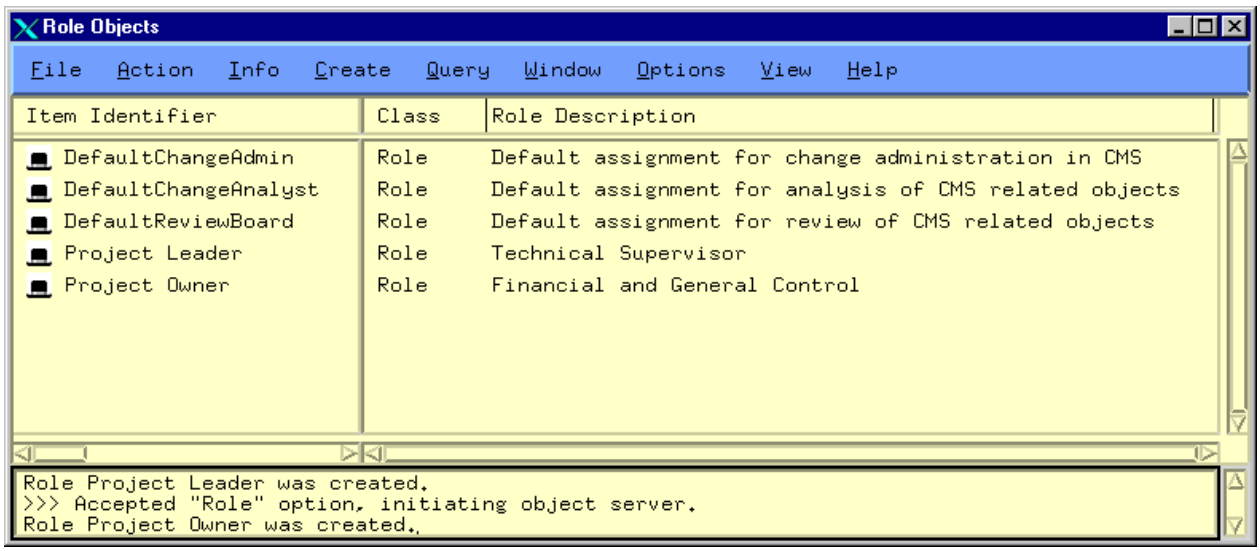
1. Create the functional user groups, if you need more than the Teamcenter Enterprise standard ones:

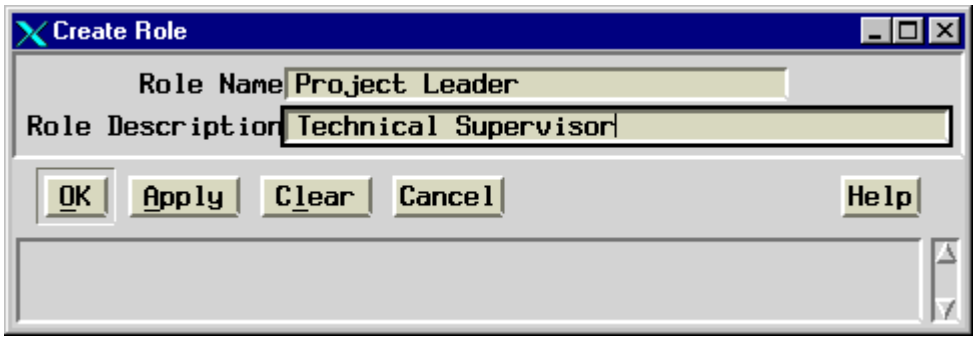




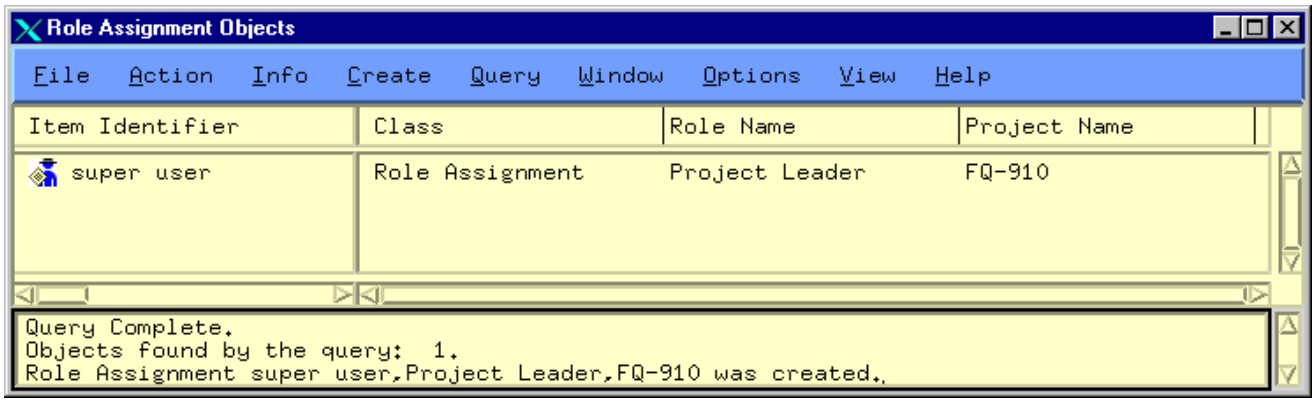
2. Assign users to their functional groups by drag and drop.

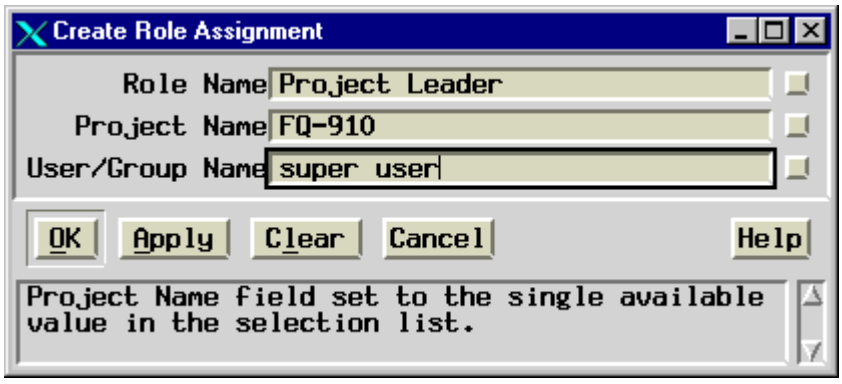
3. Create the necessary Roles:





4. Assign users to roles for each project:





3 Location Selection Rules

3.1 Physical aspects

Location selection rules are the tool to manage disk space as far as Vault and Team Locations are concerned.

The following performance rules apply to any type of mass disk storage, thus in Teamcenter Enterprise to Vault Locations.

1. Disks are physical devices and have a volume limit. Any trial to overload that size generates a OS level error.
2. Directory organization on OS level is not intended to contain unlimited amount of files. Localization of a file on OS level in a directory becomes tremendously slow, if a directory contains more than 2000-10,000 single names (files or subdirectories). This number may depend on platform and system overall performance.
3. Backup utilities run easily into timeout, if they care about directories with more some thousands of files.

The following rules about Vault Locations are Teamcenter Enterprise specific:

- If a file gets checked out and checked in again, this new copy has to go into the same Vault Location (i.e. same directory) as was his predecessor, independent from the fact, whether the Vault is defined as “replace” or “no replace”. There is no way around this[7]. Thus you never should have any Vault Location on a disk, which physically is “full”. This would prevent you from any further check out / check in.[8]

The concept of "Selection Rules" allows even to have distinct Vault Locations for a Vault respecting:

- File type or application
- Geographic location
- Platform
- Project
- …

A known **sample** for such application dependent "Selection Rules"

- We have common Vault Locations for all application and use randomized file names in the Vaults, to avoid file name conflicts.
- One application (i.e. a CAD system) may need full access in read mode to all vaulted files, and it has to be through original application generated file names. Thus we create one (or later more) specific Vault Location(s), which accepts only the relative file type. This location will not user randomized file names, because the relative application will not create conflicting names. And all other Vault Locations should not accept file of this specific format.

3.3 The "Location Selection Rule"

3.3.1 Arguments

Details see "OMF Administrator's Manual", chapter 12.

- Condition
- Class / Inherit
- User/Vault/External System
- Location/Area
- Host
- Priority

Following text is copied from this manual:

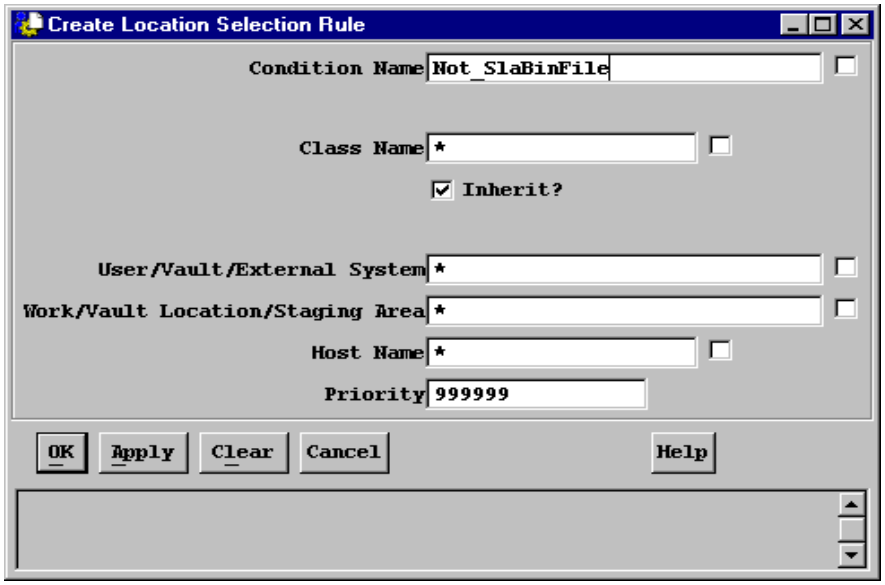
*If **Condition** (obj,location,host) and object in **Class***
*and the target is **User/Vault/External System***
*then this **Location/Staging Area** on this **Host** has this **Priority**.*

Table 12-5. Create Location Selection Rule Dialog Window Fields

Field	Description
Condition Name	Specifies the name of the condition to use on the object. If the result is TRUE, the rule applies. This field must contain a value. Click the button to the right of the field to display a list of values. The condition must be a "3 three argument condition"; arguments are the object (= file to send), location (= Vault Location to accept the file) and Host (= from which the Location will be accessed).
Class Name	Specifies the name of the class to which this rule applies. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values.
Inherit?	Specifies whether the rule applies to all subclasses of the class named in the Class Name field.
User/Vault/External System	Specifies the name of the user, vault, or external system you are writing the location selection rule for. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values.
Work/Vault Location/Staging Area	Specifies the work location, vault location, or staging area to be associated with the user, vault, or external system. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values.

Host Name	Specifies the host name. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values. With the use of NFS, you can mount the same file system from several machines. This allows applications to access the files as if they were mounted on a local disk. The host attribute recognizes this NFS capability and helps you designate the machines that write to the controlled directories. Having several NFS mounted Vault Locations accessible by all servers, you can still prefer, that the default for each server is the Location physically located on that server.
Priority	Specifies the priority for this rule. The lower the priority number, the higher the priority. This field must contain a value.

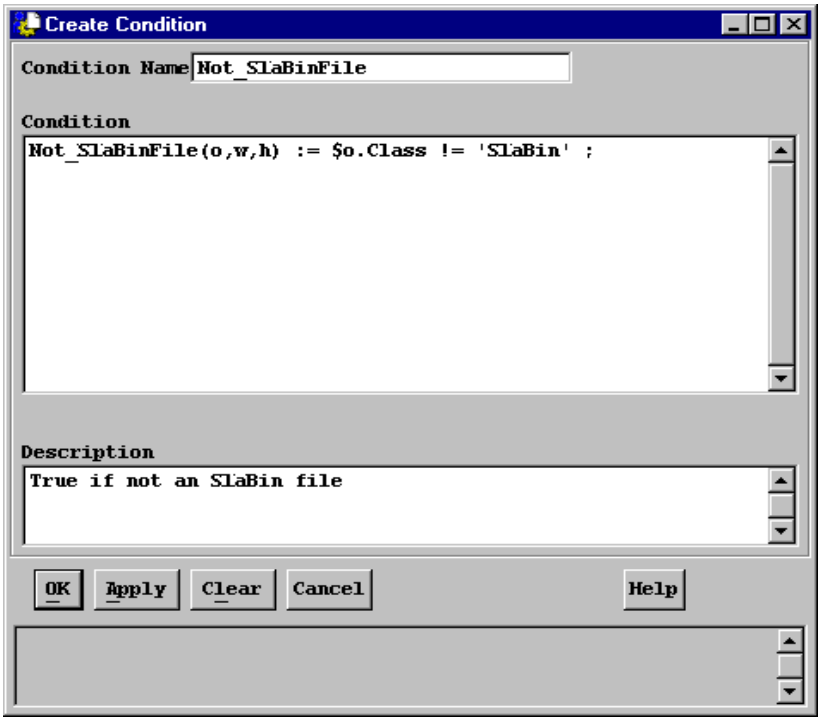
3.3.2 Samples

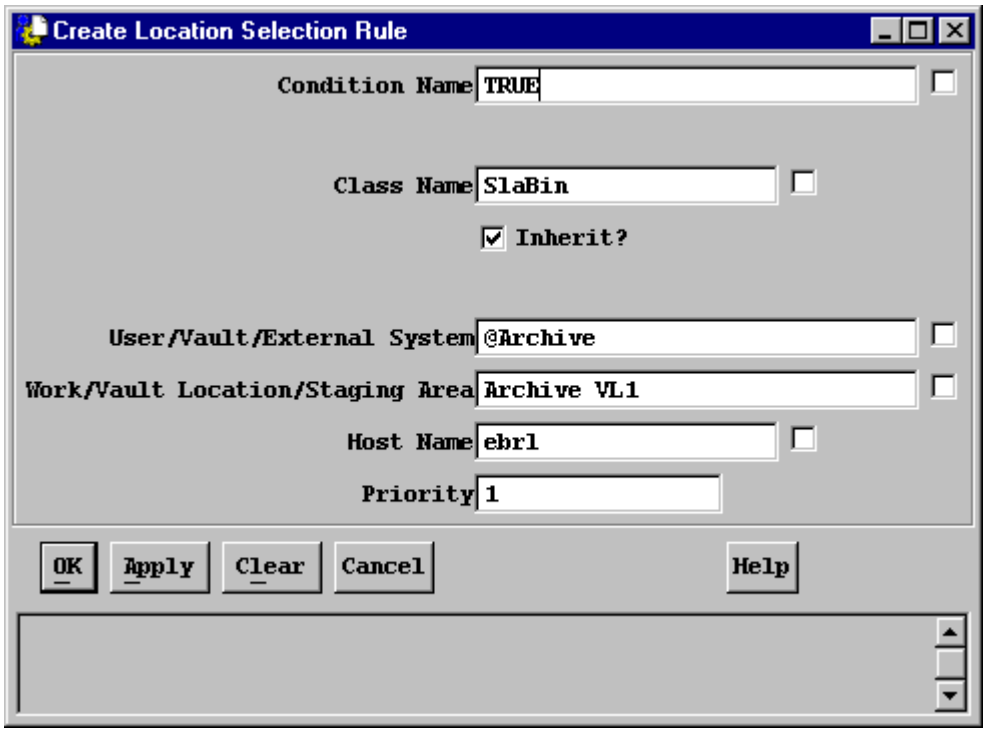


Create a default common "Selection Rule" for all Vault Locations, but not for "SlaBin" files:

Be aware, that this rule allows checking in/transfer into any Vault Location and any users Work Location. (See details on the list of rules at the end of this chapter).

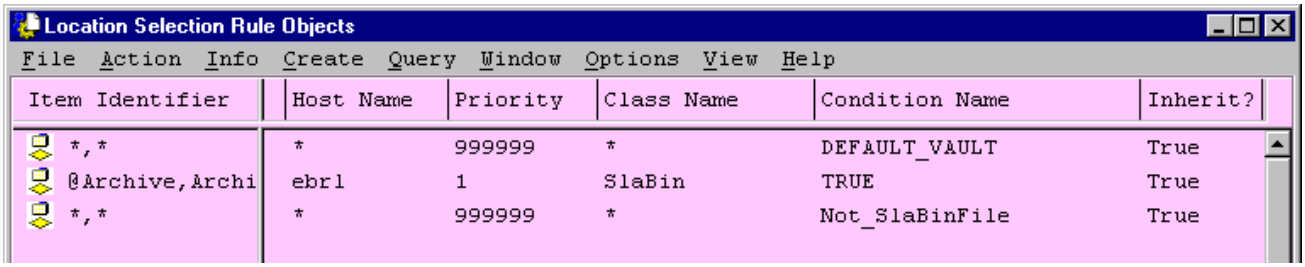
While the Condition was defined as:





This is a a rule just for "SlaBin" type files, going to "Archive VL1" if accessed from Host "ebrl".

Browser with List of "Location Selection Rules"



This Browser contains the two samples from above and the rule loaded by standard Teamcenter Enterprise at installation time.

This standard rule contains the misleading condition name "DEFAULT_VAULT". Indeed the rule is not intended for Vaults, but rather allows file transfers to other users Work Location, if that Work Location has set the "Accept Transfer" flag. (The condition checks this flag; but Vault Locations do not have this attribute, while Work Locations have it).

The third rule makes the first on more or less superfluous, because it allows transfers to any location for any file, even to any work location, except for "SlaBin" files.

Such superfluous rules may be detected in almost any environment. And they have no meaning, but they increase the number of rules to check on each menu click, thus they reduce performance.

3.4 The Vault Administrators Work

1. There is a Vault and Vault Location concept and convention to be established at start of a Teamcenter Enterprise instance. This concept needs to include the necessary "Selection Rules" for all Work Locations, Common General Vault Location rule and rule for each Vault's default Location in this starting state. Later conception changes have to include the relative rules.

2. An administrator is responsible to watch file spaces and number of files in Vault Locations directories. Whenever a location gets near to full (80%) or near to a max number of files, he creates a new additional location for that Vault.

3. For each new default Vault Location the rule defining with high priority the Vault's default location has to be updated, so that it points to the new default.

4 Notification Rules

Notification rules define:

Who gets a notification upon which **event** on a **class** object and under which **condition**?

4.1.1 Arguments

Details see "OMF Administrator's Manual", chapter 12.

- Condition
- Class / Inherit
- Participant (mail recipient)
- Event

Following text is copied from this manual:

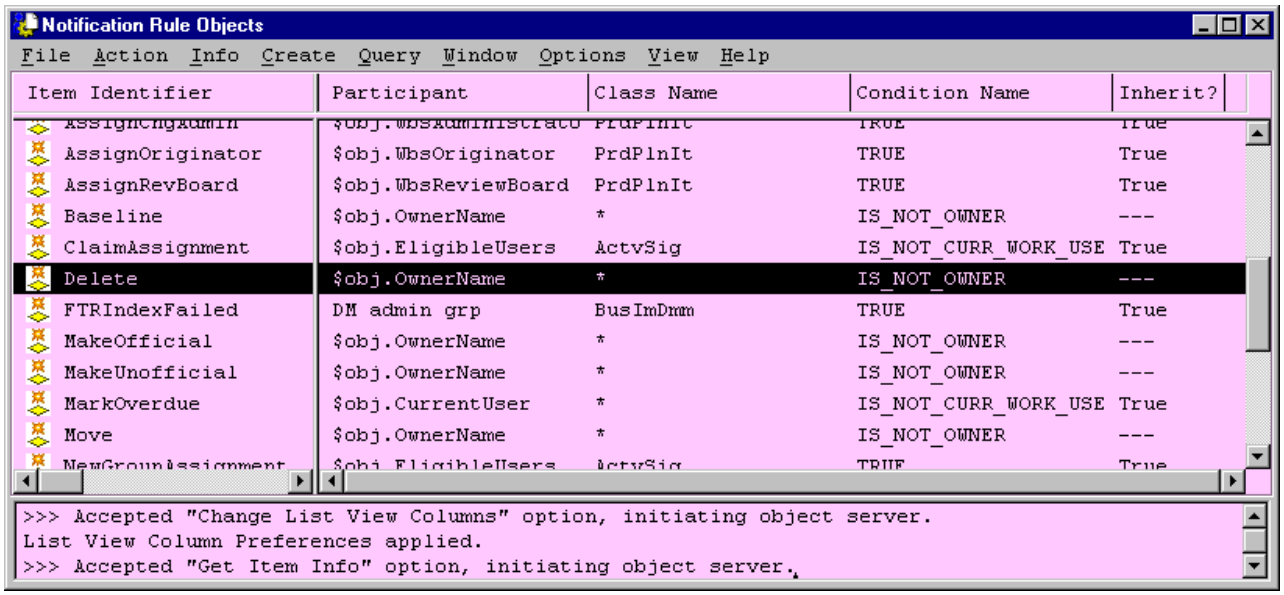
*When **Event** occurs to object in **Class** if **Condition** (obj,env),*
*then send mail to **Participant**.*

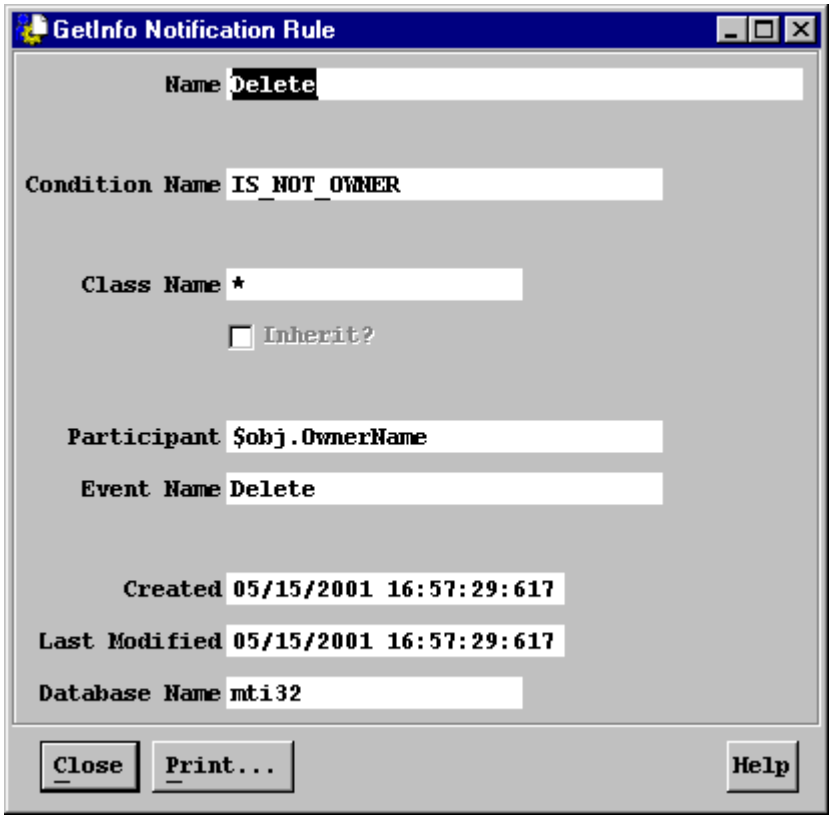
Table 12-8. Create Notification Rule Dialog Window Fields

Field	Description
Condition Name	Specifies the name of the condition to use on the object. If the result is TRUE, the rule applies. This field must contain a value. Click the button to the right of the field to display a list of values. The same type of conditions apply as for "Message Access Rules", i.e. conditions with two arguments: Object and environment.
Class Name	Specifies the name of the class to which this rule applies. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values.
Inherit?	Specifies whether the rule applies to all derived classes of the class named in the Class Name field.
Participant	Specifies the person for which the rule is intended. You can enter the name of a user, group, or role. You can also enter the name of an attribute of the object itself, if the attribute contains a participant name (user, group, or role) or list of participant names. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values. The Participant here is the recipient of the mail, not the actor on the object.
Event Name	Specifies the name of the event for which the rule is intended. You can use wildcard values in this field. This field must contain a value. Click the button to the right of the field to display a list of values.

4.1.2 Samples

The following is part of the list of standard Teamcenter Enterprise Notification rules:





This is the GetInfo dialog of the highlighted rule in the browser above.

It defines, that the Owner of an object get an eMail notification, if somebody else deletes it (this needs extra right or can be done by a super user).