Requirements Specifications for ORBIT Access Control

Author: Saswati Swami

I. Policy Identification

A Role-Based Access Control (RBAC) policy has been adopted for this project. This is because an RBAC system enables users to carry out a broad range of authorized operations, and provides great flexibility and breadth of application. System administrators can both statically and dynamically regulate user's actions through the establishment and definition of roles, role hierarchies, relationships, and constraints. Once such an RBAC system is established, the principal administrative actions are the granting and revoking of users into and out of roles. Thus this policy is devoid of the complexity that is associated with the more conventional and less intuitive process of attempting to administer lower-level access control mechanisms directly (e.g. access control lists (ACLs), capabilities, or type enforcement entities) on an object-by-object basis.

This document describes the RBAC framework for the ORBIT project, which is based on the $RBAC_3$ model as in [4]. The framework lists the resources, the Roles that have been identified for ORBIT, the authorization rights for each Role and the use cases. A thorough analysis of the system has been done to identify all possible use cases, and all the use cases preconditions to decide what authorization rights are needed for each Role. Also, a *Role Reference Matrix* has been used to identify the correlation between Roles and resources, the mutually exclusive Roles, and the Role hierarchies.

The following convention has been followed all throughout this document:

1. RG.x: numbering scheme for generic requirements
2. RG.x.x: numbering scheme for generic sub-requirements
3. RS.x: numbering scheme for ORBIT RBAC Framework
4. RS.x.x: numbering scheme for ORBIT RBAC Framework

II Generic Role-Based Access Control Requirements

This section lists the specifications that constitute the core requirements that have to be supported by all systems that incorporate and implement a Role-based Access Control policy. The ORBIT access control system will have support for all of these.

RG.1 *Users and Roles*

[RG.1.1] Access decisions are based on the roles that individual users have as part of an organization. Users take on assigned roles.

[RG.1.2] Access rights are grouped by role name, and the use of resources is restricted to individuals authorized to assume the associated role.

[RG.1.3] Users are granted memberships into roles based on their competencies and responsibilities in the organization.

[RG.1.4] The operations that a user is permitted to perform are based on the user's role.

[RG.1.5] User membership into roles can be revoked easily and new memberships established as job assignments dictate.

[RG.1.6] Role associations can be established when new operations are instituted, and old operations can be deleted as organizational functions change and evolve.

[RG.1.7] When a user is associated with a role, the user will be given no more privilege than is necessary to perform the job in accordance with the *Principle of Least Privilege*.

RG.2 *Roles and Role Hierarchies*

[RG.2.1] Under RBAC, roles can have overlapping responsibilities and privileges; that is, users belonging to different roles may need to perform common operations.

[RG.2.2] All employees may perform some general operations.

[RG.2.3] Role hierarchies can be established to provide for different role flows. A role hierarchy will define the roles that have unique attributes and that may contain other roles; that is one role may implicitly include the operations that are associated with another role.

[RG.2.4] Mutually exclusive roles to be clearly defined.

[RG.2.5] It has to be ensured that the role in which the user is gaining membership is not mutually exclusive with another role for which the user already possesses membership.

[RG.2.6] When operations overlap, hierarchies of roles can be established.

[RG.2.7] Roles are activated statically and dynamically as appropriate.

[RG.2.8] Roles can only be transferred or delegated using strict sign-offs and procedures.

[RG.2.9] A central primary security administrator or a primary project leader at the highest level should be able to manage the roles centrally.

RG.3 *Roles and Operations*

[RG.3.1] Organizations can establish the rules for the association of operations with roles.

[RG.3.2] Operations can also be specified in a manner that can be used in the demonstration and enforcement of laws or regulations.

[RG.3.3] An operation represents a unit of control that can be referenced by an individual role, subject to regulatory constraints within the RBAC framework.

[RG.3.4] The RBAC framework provides administrators with the capability to regulate who can perform what actions, when, from where, in what order, and in some cases under what relational circumstances:

[RG.3.4.1] Only those operations that need to be performed by members of a role are granted to a role.

[RG.3.4.2] Granting of user membership to roles can be limited.

[RG.3.4.3] Some roles can only be occupied by a certain number of employees at any given period of time. A user can become a new member of a role as long as the number of members allowed for the role is not exceeded.

RG.4 *RBAC for Distributed Systems*

[RG.4.1] Administrator responsibilities will be divided among central and local protection domains.

[RG.4.2] Protection policies that commonly affect everyone will be defined at the central level.

[RG.4.3] Protection issues of local concern will be decided at the unit level. e.g. granting and revoking of membership into specific roles may be specified at the local level.

III Specific ORBIT Role-Based Access Control Requirements

The ORBIT RBAC Framework constitutes the various defined Roles that will be assigned to the various system users, the identification of resources for which access control will be in effect, the authorizations for each resource that are granted to each role, and the hierarchical relationship between the various roles.

The requirements specific to the ORBIT RBAC Framework are listed in this section. The Framework will support these requirements in addition to the ones listed in the previous section.

[RS.1] The ORBIT users can be assigned to any one or more of these Roles. Some of these Roles have a hierarchical relation between them and also some are mutually exclusive to each other. The following Roles have been identified in the ORBIT project:

[RS.1.1] Project Role (PR): A project can consist of any number of experiments. A Project will have many members called users who fulfill the User Role. The users need to be explicitly granted membership to a Project before being able to run experiments for that Project. Each project will have a Project Lead.

[RS.1.2] User (Experimenter) Role (UR): The User Role signifies those ORBIT users who will be allowed to run experiments on ORBIT. A user is not restricted to a single project and can hold membership to many projects for each of which the user will be permitted to run experiments. Thus, the User Role will have a many-to-many relationship with the Project Role and a one-to-many relationship with the Project Membership Role.

The *constraints* associated with this role are:

[RS.1.2.1] A user will be granted membership to a Project only once. So, the maximum number of Project Memberships that the user will be granted is equal to the number of projects.

[RS.1.3] Project Membership Role (PMR): Users need to be explicitly granted membership of a Project before the users can run experiments for the Project. This granting of project membership is fulfilled by this Role. A user can have the membership of several Projects and so this Role has a many-to-one relationship with the User Role.

The *constraints* associated with this role are:

[RS.1.3.1] The User Role is a *prerequisite role* for the Project Membership Role as the Project Membership Role can only be granted to those who have the User Role.

[RS.1.4] Project Lead Role (PLR): A Project Lead will have supervisory permissions on users (remove an user from a project, add an user to a project, etc.) and also will itself be a user. This role will have a one-to-many relationship with the User Role and possess all the authorizations that are granted to the User Role in addition to authorizations of its own. This role will have a one-to-one relationship with the Project Role. The Project Lead can also add / remove resources to / from the Project. Additionally, the Project Lead will can temporarily delegate its responsibilities to the delegated project lead and also decide what responsibilities to grant. So, this role can have a one-to-many relationship with the Delegated Project Role.

The *constraints* associated with this role are:
[RS.1.4.1] A project can have only one Project Lead at a time. So, the maximum number of Project Leads at any time is less than or equal to the number of Projects.

[RS.1.4.2] The User Role and the Project Membership Role are *prerequisite roles* for this role.

[RS.1.5] Delegated Project Lead Role (DPLR): A Project Lead may temporarily permit a project member to assume the responsibilities of a Project Lead. This project member will then assume the Delegated Project Lead Role. The Project Lead will have the freedom to decide the extent of the rights granted to this Role. This Role will have a one-to-one relationship with the Project Role.

The *constraints* associated with this role are:
[RS.1.5.1] This Role is temporarily granted for a specified amount of time at the end of which all the granted rights and privileges are revoked.

[RS.1.5.2] This Role can be granted to one/many user(s) at a time for any project.

[RS.1.5.3] The User Role and the Project Membership Role are *prerequisite roles* for this role.

[RS.1.6] Administrator Role (AR): The Administrator Role will consist of system maintenance, add/remove developers, user authentication, creating and deleting new users, performing authorization functions, setting session time parameters, setting refresh time parameters, adding new roles, deleting existing roles, updating existing roles etc. The Administrator can temporarily delegate its responsibilities to a person who fulfils the Delegated Administrator Role. So, this role has a one-to-one relationship with the Delegated Administrator Role.

The *constraints* associated with this role are:
[RS.1.6.1] There will be only one Administrator at any time.

[RS.1.7] Delegated Administrator Role (DAR): The Administrator can temporarily permit another person to assume the responsibilities of the Administrator for a specified period of time. At the end of the specified period, all rights and privileges will be revoked. The Administrator will decide the extent of the rights granted to this role.

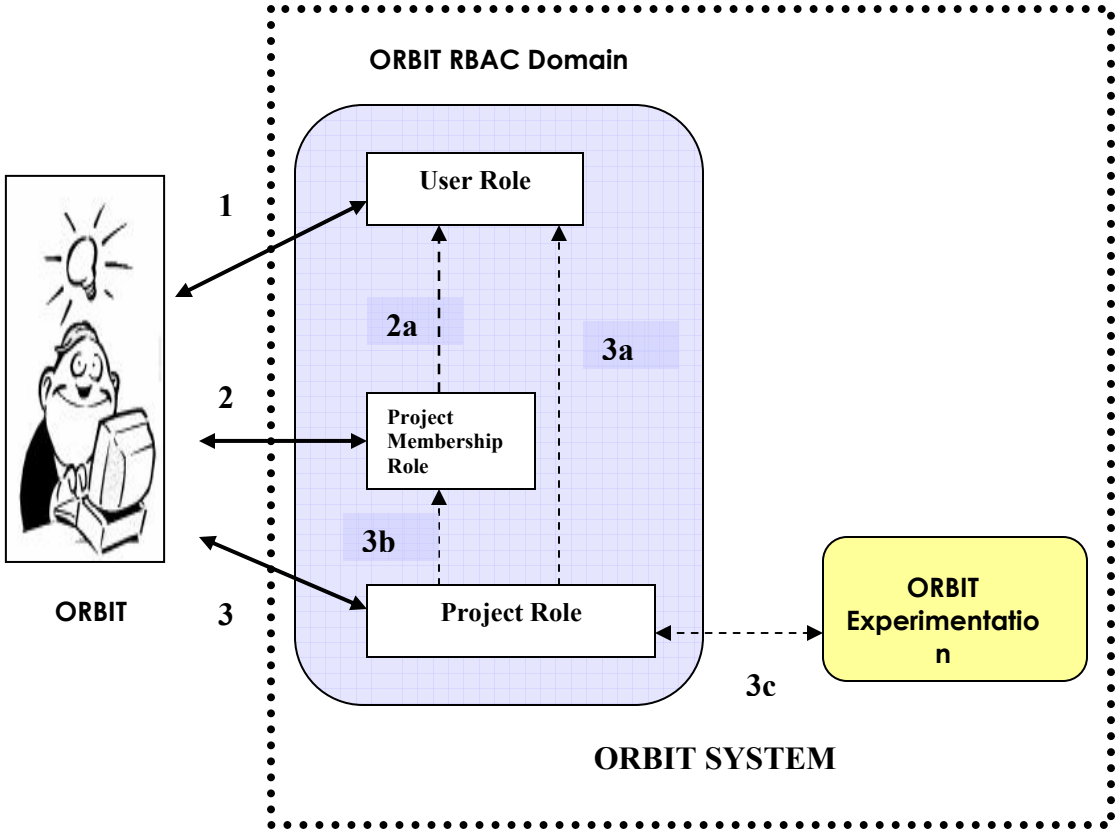
S.1.6.1] There can be more than one Delegated Administrator at any time.

[RS.1.8] Developer Role (DR): This role will authorize a separate group of users to enhance, develop, modify and test ORBIT code.

[RS.2] The following relationship exists between the different roles:

	PR	UR	PMR	PLR	DPLR	AR	DAR	DR
PR	*Self*	*Many-to-many*	*None*	*One-to-one*	*One-to-one*	*None*	*None*	*None*
UR	*Many-to-many*	*Self*	*One-to-many*	*Many-to-one, child, Prerequisite*	*Many-to-one, child, Pre-requisite*	*None*	*None*	*None*
PMR	*None*	*Many-to-one*	*Self*	*Prerequisite*	*Pre-requisite*	*None*	*None*	*None*
PLR	*One-to-one*	*One-to-many, Parent*	*One-to-many*	*Self*	*Delegates*	*None*	*None*	*None*
DPLR	*One-to-one*	*One-to-many, Parent*	*One-to-many*	*Delegated*	*Self*	*None*	*None*	*None*
AR	*None*	*None*	*None*	*None*	*None*	*Self*	*Delegates*	*None*
DAR	*None*	*None*	*None*	*None*	*None*	*Delegated*	*Self*	*None*
DR	*None*	*None*	*None*	*None*	*None*	*None*	*None*	*Self*

[RS.3] ORBIT User interaction with the User Role, the Project Role and the Project Membership Roles are diagrammatically explained below.



1. User requested and registered as an ORBIT user.
2. User granted membership to requested projects.

2a. Authorization policy verifies that user has been registered as an ORBIT user.

3. User starts running experiments on ORBIT.

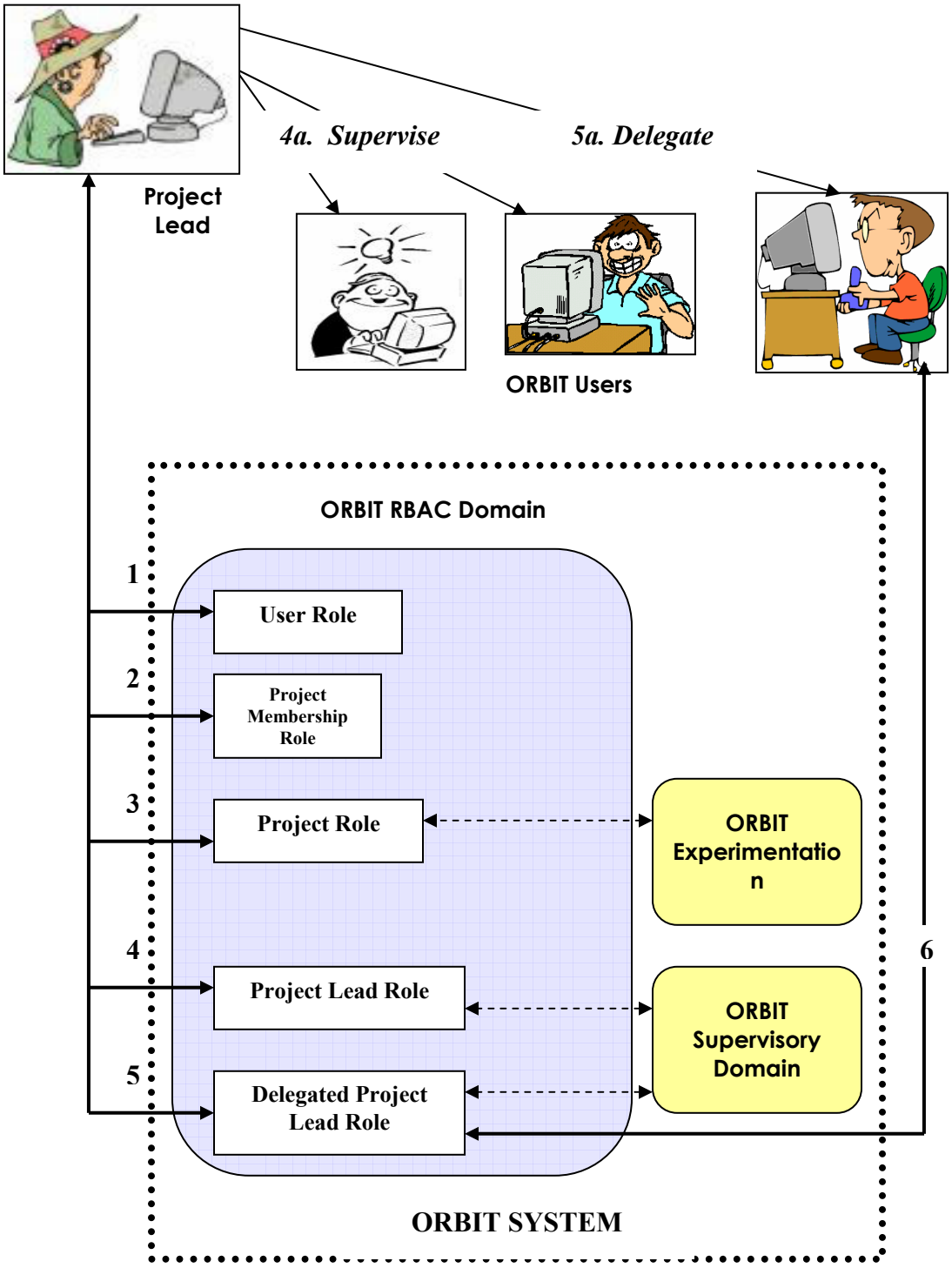
3a. Authorization policy verifies that user has been registered as an ORBIT user.

3b. Authorization policy verifies that user has been granted membership to the project for running experiments.

3c. ORBIT RBAC coordinates the resource allocation & authorization as per the pre-defined resource allocation policy.

Fig. 1: User interaction with ORBIT RBAC

[RS.4] Project Lead interaction with the User Role, the Project Role, the Project Membership Role, the Project Lead Role and the Delegated Project Lead Role are diagrammatically explained below.



1. Project Lead requested and registered as an ORBIT user.
2. Project Lead granted membership to requested projects.
3. Project Lead authorized to run experiments on ORBIT.
4. Project Lead registers as lead for a project.

4a. RBAC grants supervisory rights over ORBIT users for the project.

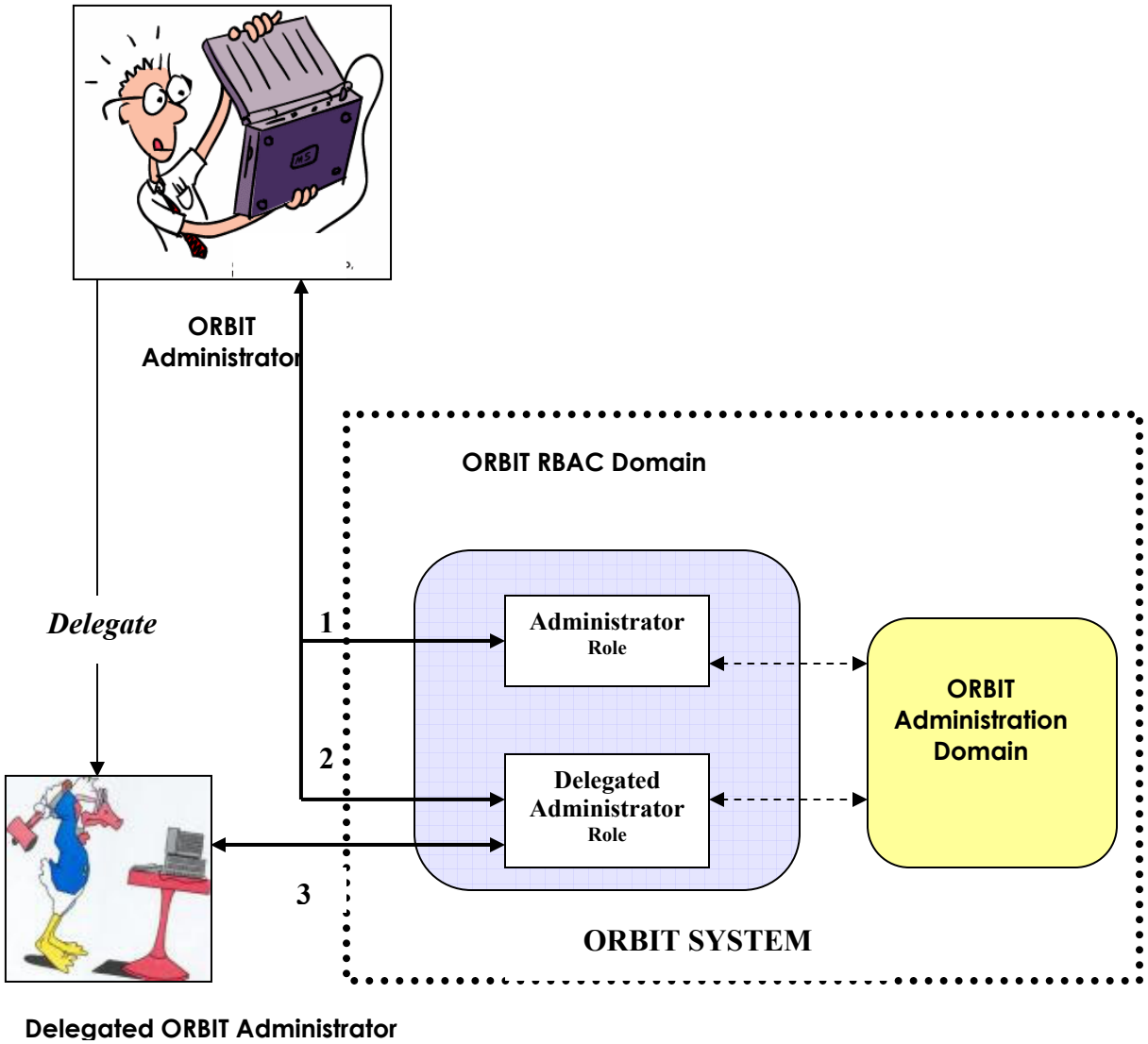
5. Project Lead registers an ORBIT User as its delegate.

5a. RBAC authorizes Project Lead to delegate

6. Delegated Project Lead registers and granted supervisory rights by RBAC.

Fig. 2: Project Lead interaction with ORBIT RBAC

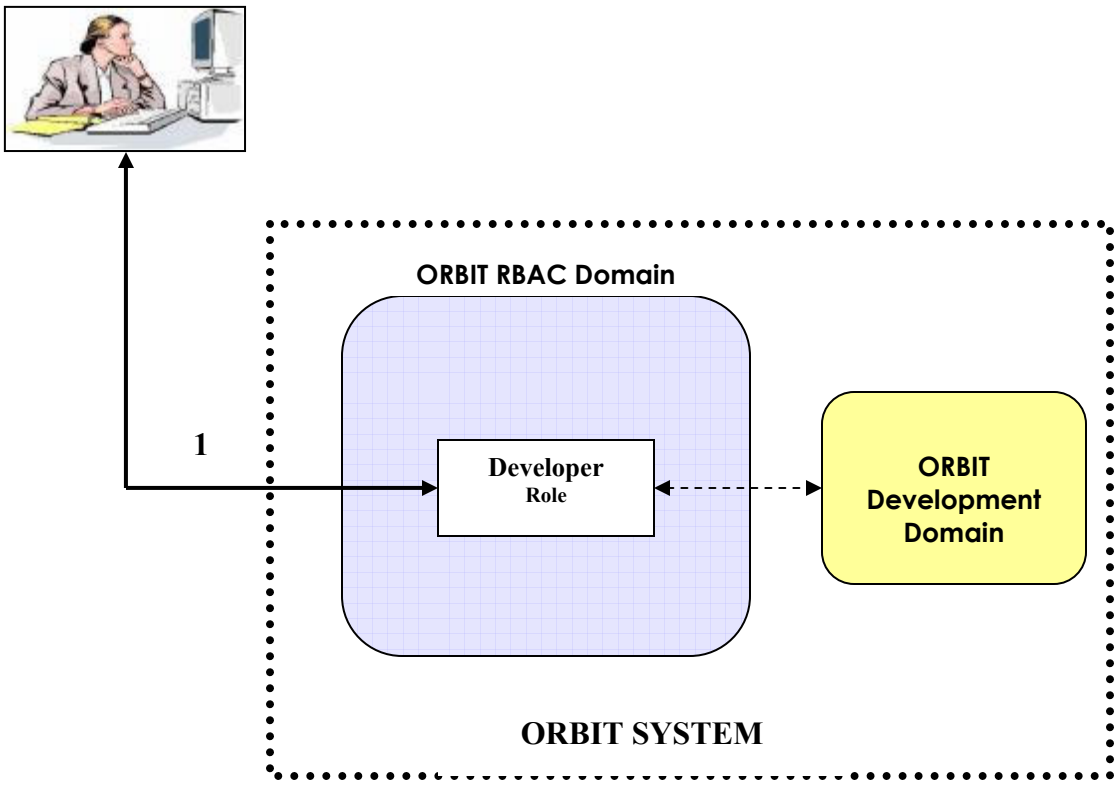
[RS.5] ORBIT Administrator interaction with the Administrator Role and the Delegated Administrator Roles are diagrammatically explained below.



1. The Administrator gets registered as ORBIT Administrator from ORBIT RBAC. The Administrator is then allowed to access the ORBIT Administrative Domain.
2. The ORBIT Administrator registers a delegate with RBAC and is authorized by RBAC to grant certain rights.
3. The Delegated ORBIT Administrator registers itself with RBAC and is allowed access to the ORBIT Administrative domain.

Fig. 3: Administrator interaction with ORBIT RBAC

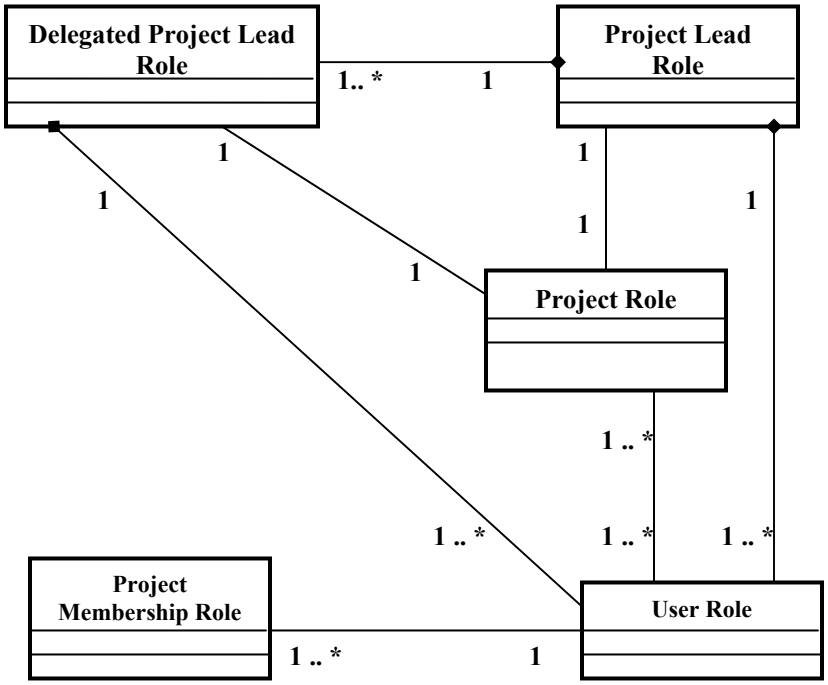
[RS.6] ORBIT Developer interaction with the RBAC is diagrammatically explained below.

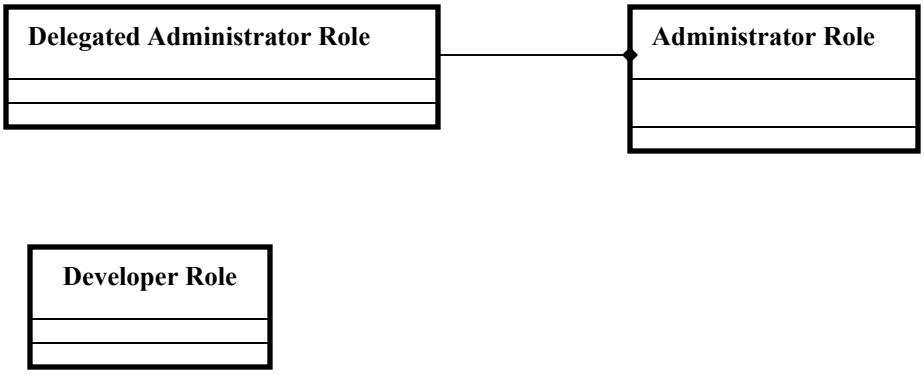


1. The Developer gets registered as ORBIT developer from the ORBIT RBAC. The ORBIT developer is then allowed access to the ORBIT Development Domain and can develop, modify, test code.

Fig. 4: Developer interaction with ORBIT RBAC

[RS.7] The relationship between the roles is expressed diagrammatically as follows:





[RS.5] A Role Reference Matrix has been defined to correlate the ORBIT resources and role authorizations.

	UR	PR	PMR	PLR	DPLR	AR	DAR	DR
iDB read	Granted	Granted	Not granted	Granted	Granted	Granted	Granted	Not granted
iDB delete	Not Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
iDB write	Not Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
iDB update	Not Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
iDB create	Not Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
eEDB access	Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
File System (files)	Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Granted to modify & create files
Instrumentation (Chassis Manager Service, Sniffer)	Not granted	Not granted	Not Granted	Not Granted	Not granted	Granted	Granted	Not granted
Noise Generator Access	Not granted	Not granted	Not Granted	Not Granted	Not Granted	Granted	Granted	Not granted
Grid Authentication	Not granted	Not granted	Not Granted	Not Granted	Not granted	Granted	Granted	Not granted
Internal Servers	Not granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
Remote Data Acquisition	Not Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
Application#x	Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
SandBox#	Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
Grid	Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted
Network Devices	Granted	Granted	Not Granted	Granted	Granted	Granted	Granted	Not granted

IV Usage Cases

In this section, both the core and the specific requirements mentioned in the previous section will be mapped to the ORBIT RBAC Framework through use-cases.

Table IV.1

Title	**Login User to ORBIT**
Roles	**Project Role, User Role, Project Membership Role**
Requirement#	
Pre-Conditions	**ORBIT system is up, the user attempting login has a valid user account**
Post-Conditions	**The user is allowed to login successfully and allowed to navigate to experiment start directories**
Purpose	**User attempts to login as User wants to access his account on ORBIT.**
Description	**User fills in login id, password. ORBIT RBAC checks whether the user has membership of any existing Project.**

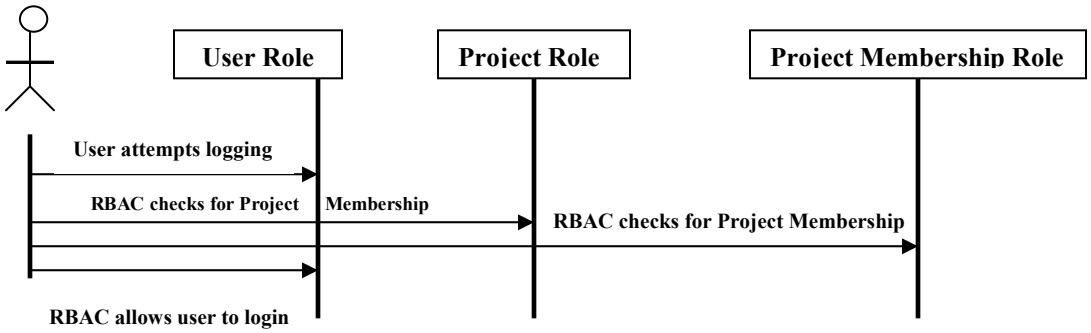
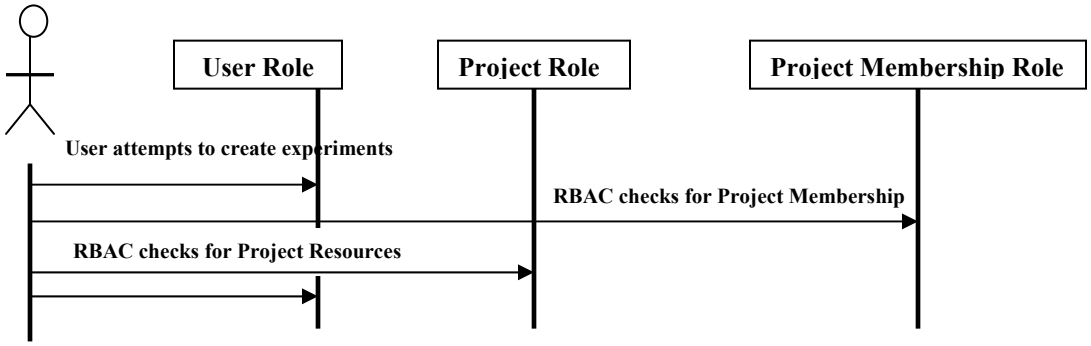


Table IV.2

Title	**Create Experiments on ORBIT**
Roles	**Project Role, User Role, Project Membership Role**
Requirement#	
Pre-Conditions	**ORBIT system is up, the user has already logged in successfully.**
Post-Conditions	**The user is able to successfully create and run the experiments.**
Purpose	**User wants to execute experiments on ORBIT.**
Description	**User fills in the Project name. ORBIT RBAC allows user to create experiments only for a Project of which the user is a member. Also, it allows the use of only those resources in the experiment for which both the user and the Project have the necessary authorizations.**



RBAC allows user to create experiments

Table IV.3

Title	Run Experiments on ORBIT
Roles	Project Role, User Role, Project Membership Role
Requirement#	
Pre-Conditions	ORBIT system is up, the user has already logged in successfully.
Post-Conditions	The user is able to successfully run the experiments.
Purpose	User wants to execute experiments on ORBIT.
Description	User fills in the Project name. ORBIT RBAC allows user to run experiments for the specified Project if the user has membership of that Project. Also, it allows the user to run only those experiments for which the user has the required authorizations.

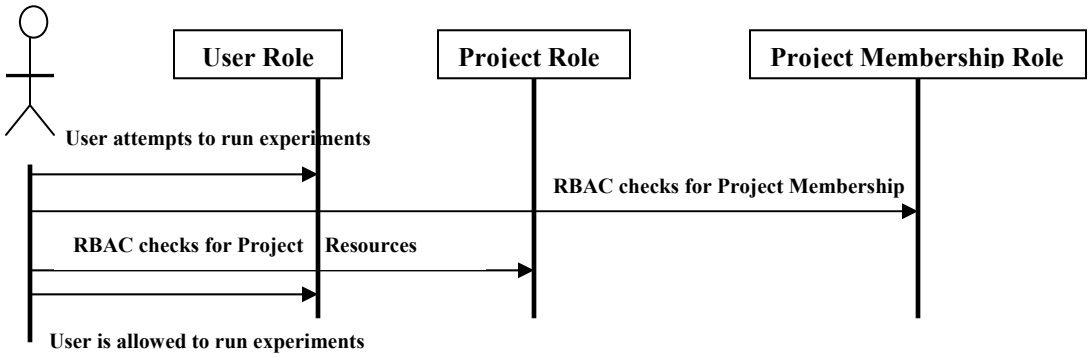
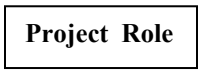


Table IV.4

Title	View measurement Data
Roles	Project Role, User Role
Requirement#	
Pre-Conditions	ORBIT system is up; the user has already logged in successfully and has membership of that Project.
Post-Conditions	The user is able to successfully view measurement data.
Purpose	User wants to view measurement data of all experiments in the Project.
Description	User fills in the Project name. ORBIT RBAC allows user to view measurement data for all experiments that constitute that Project. Viewer access for other Projects is not allowed.



User attempts to view data

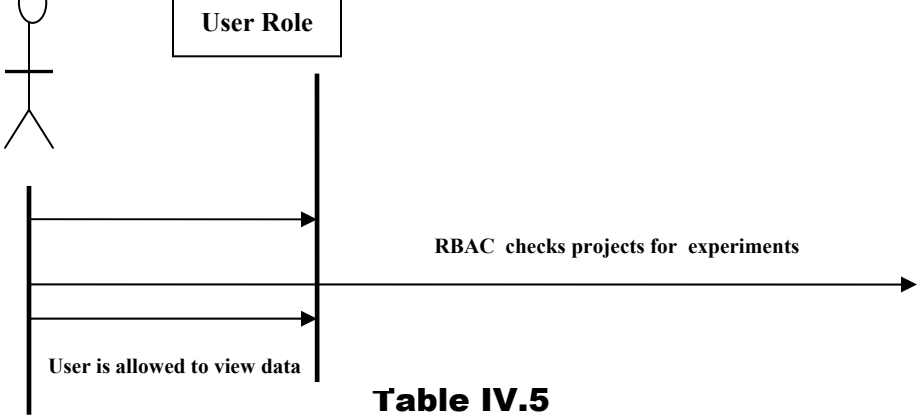


Table IV.5

Title	Modify measurement Data
Roles	Project Role, User Role
Requirement#	
Pre-Conditions	ORBIT system is up; the user has already logged in successfully and has membership of that Project.
Post-Conditions	The user is able to successfully modify measurement data for experiments that have been created by the user.
Purpose	User wants to modify measurement data of all experiments in the Project.
Description	User fills in the Project name. ORBIT RBAC allows user to modify measurement data only for those experiments that are constituents of the specified Project and also have been created by the user. For all other experiments for the specified Project, user only has read access.

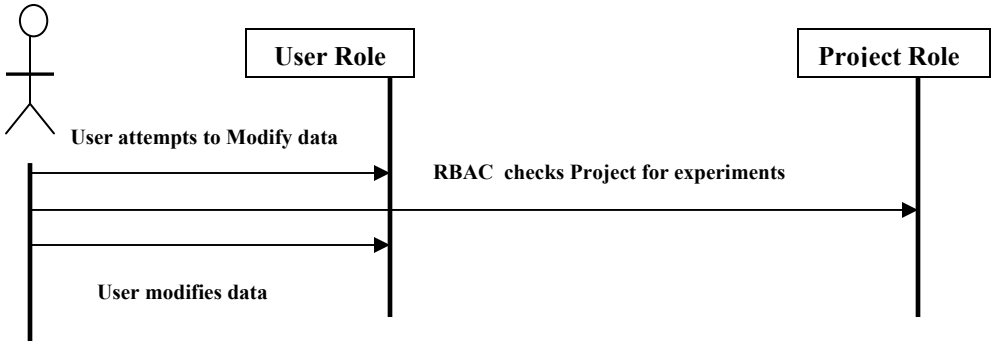


Table IV.6

Title	Add new User to Project
Roles	Project Membership Role, Project Role, Project Lead Role
Requirement#	
Pre-Conditions	ORBIT system is up; the user has requested the Project Lead for membership to one or more Projects.
Post-Conditions	The user is able to successfully run experiments for the Project in which the user has been granted membership.
Purpose	Grant access to new User.
Description	Project Lead grants membership to the user. The Project Lead also specifies the experiments that the user is authorized to run, the resources that will be taken away from the user on timeout, the no. of experiments that the user can run simultaneously, the resources to which the user will have access to and the type of access

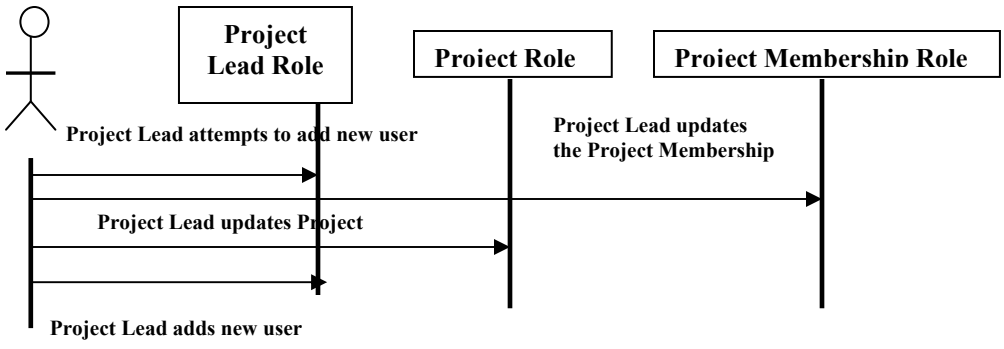


Table IV.7

Title	Remove User from Project
Roles	Project Role, Project Role, Project Lead Role
Requirement#	
Pre-Conditions	ORBIT system is up; and the Project Lead has successfully logged into ORBIT.
Post-Conditions	The user is not able to run experiments for the Project.
Purpose	Remove User access.
Description	Project Lead disables all membership authorizations for the user.

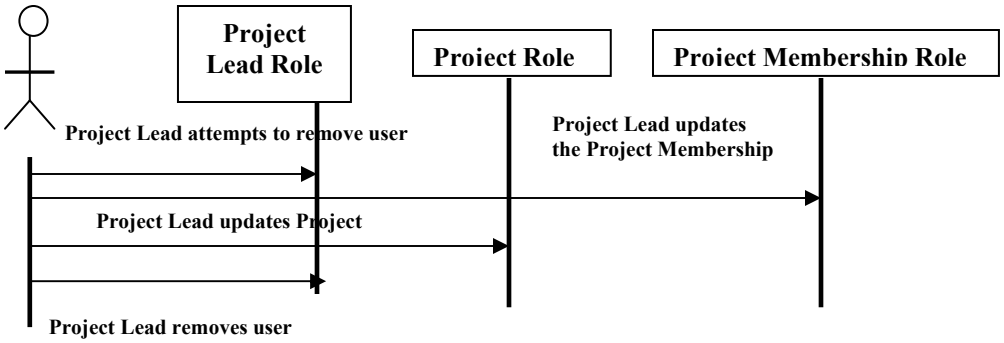


Table IV.8

Title	Add new resources to the Project
Roles	User Role / Project Lead Role, Administrator Role
Requirement#	
Pre-Conditions	ORBIT system is up; and the Project Lead has successfully logged into ORBIT.
Post-Conditions	The Project Lead successfully adds new resources, which is then available to authorized users.
Purpose	Making new resources available to the Project
Description	Project Lead interacts with the Administrator to update the resource authorization list for the Project. The Project Lead then allocates these resources to the users over which the Project Lead has supervisory permissions. The authorized Users can then update existing experiments or create new experiments that utilize these new resources.

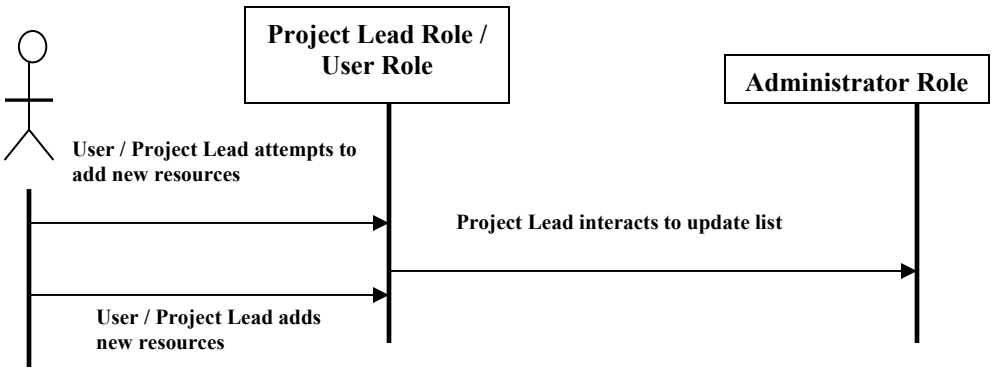


Table IV.9

Title	Delete resources from the Project
Roles	User / Project Lead Role, Administrator Role
Requirement#	
Pre-Conditions	ORBIT system is up; and the Project Lead has successfully logged into ORBIT.
Post-Conditions	The deleted resources are no longer available to the Project experiments.
Purpose	Removing redundant resources from the Project, which are then available for re-distribution among other Projects; thus preventing hogging of resources by any single project.
Description	Project Lead interacts with the Administrator to remove the resources from the authorization list for the Project. These resources can no longer be used by any experiment in the Project.

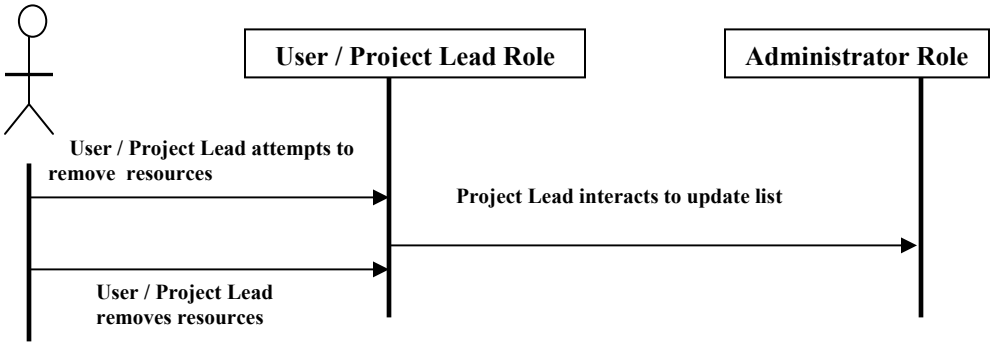


Table IV.10

Title	Request Project Membership
Roles	User Role, Project Lead Role, Project Membership Role
Requirement#	
Pre-Conditions	ORBIT system is up; and the Project Lead has successfully logged into ORBIT.
Post-Conditions	The User is granted membership to the Project that it requested.
Purpose	Granting the user membership to the Project that the user requested so that the user can start creating/running experiments.
Description	User interacts with the Project Lead requesting membership to a Project. The Project Lead updates the Project Membership list and grants membership to the user.

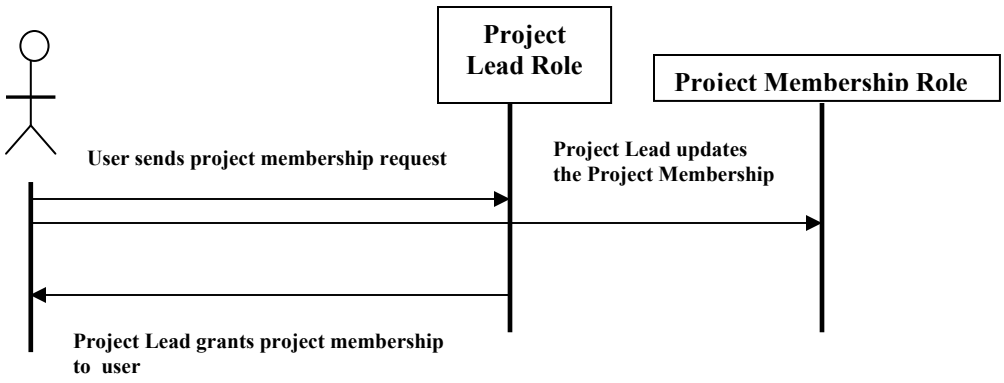


Table IV.11

Title	Add new roles
Roles	Administrator Role
Requirement#	
Pre-Conditions	ORBIT system is up.
Post-Conditions	The new Roles are available for assignment.
Purpose	Adding new Role functionality dynamically.
Description	Newly defined Roles are incorporated into ORBIT RBAC by the administrator, Role permissions set and new Role members assigned.

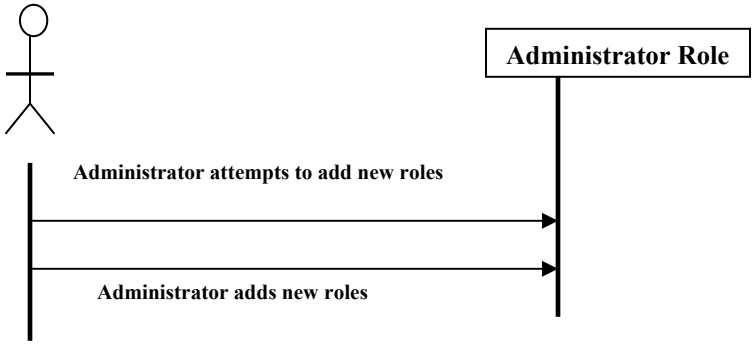


Table IV.12

Title	Delete existing roles
Roles	Administrator Role
Requirement#	
Pre-Conditions	ORBIT system is up.
Post-Conditions	The deleted roles are not present in ORBIT.
Purpose	Removing Role functionality dynamically.
Description	Specified existing Roles are removed and all such Role permissions disabled.

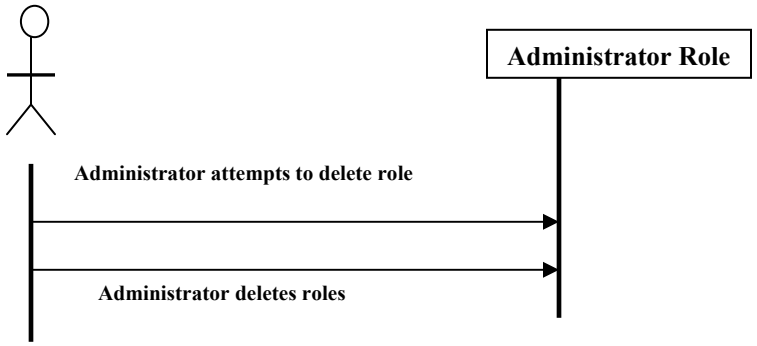


Table IV.13

Title	**Update existing Role functionality**
Roles	**Administrator Role, Project Lead Role, User Role**
Requirement#	
Pre-Conditions	**ORBIT system is up.**
Post-Conditions	**Updated Role permissions available in ORBIT.**
Purpose	**Updating Role functionality dynamically.**
Description	**The administrator updates the existing Role functionality for the specified Roles by updating the Role authorization list. The newly updated Roles function as per the new functionality.**

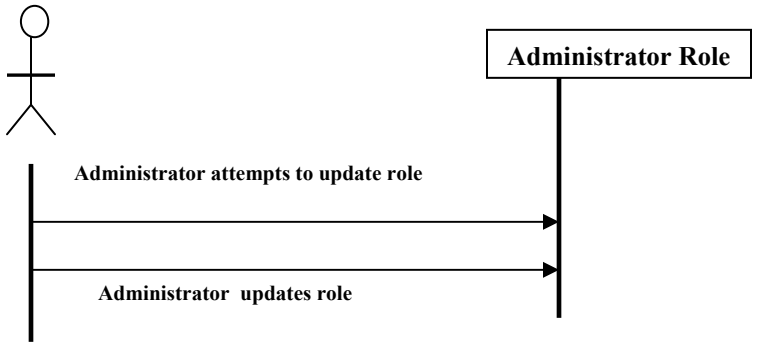


Table IV.14

Title	**Set / reset session and refresh time parameters**
Roles	**Administrator Role**
Requirement#	
Pre-Conditions	**None**
Post-Conditions	**Session Timeout and Refresh take place as per new time parameters.**
Purpose	**Dynamically set / reset session and refresh time parameters**
Description	**The administrator initially sets and then subsequently resets the session and refresh time parameters as the need arises. User sessions will timeout or be refreshed as per the new parameters.**

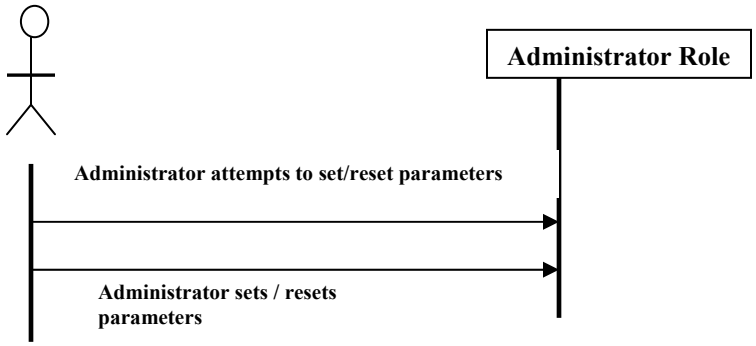


Table IV.15

Title	**Add new Developer to ORBIT**
Roles	**Administrator Role**
Requirement#	
Pre-Conditions	**None**
Post-Conditions	**Developer edits ORBIT code**
Purpose	**Adding new developers to ORBIT**
Description	**The administrator adds a new developer account. The developer logs in and then works on enhancing / maintaining ORBIT.**

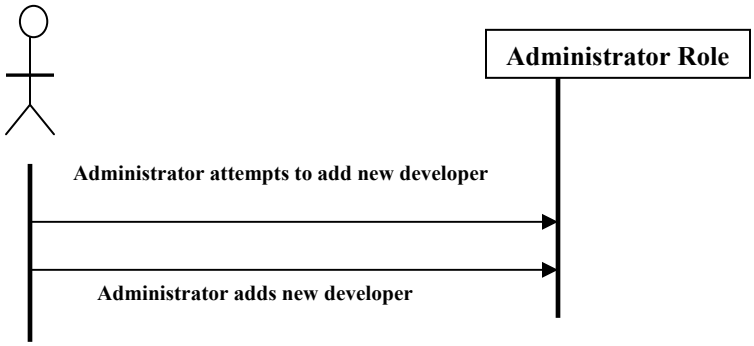


Table IV.16

Title	**Delegate Project Lead Role**
Roles	**Project Lead Role, Delegated Project Lead Role**
Requirement#	
Pre-Conditions	**There should be at least one designated Project Lead which supervises one or more users**
Post-Conditions	**Delegated Project Lead temporarily assumes the Project Lead authority.**
Purpose	**The purpose is to delegate authority temporarily in the absence of the Project Lead.**
Description	**The Project Lead decides who to delegate its authority temporarily and what to delegate. The Project Lead also decides the duration of this delegation.**

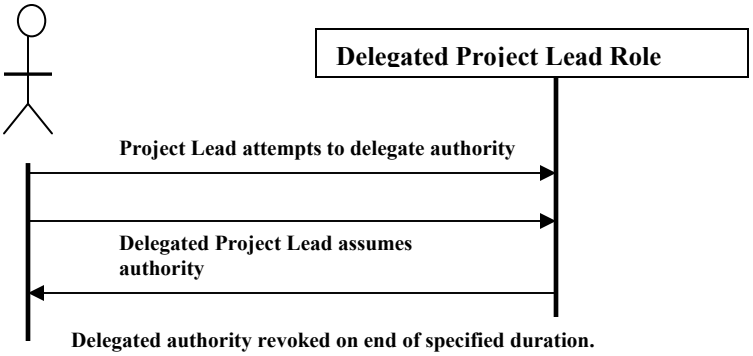
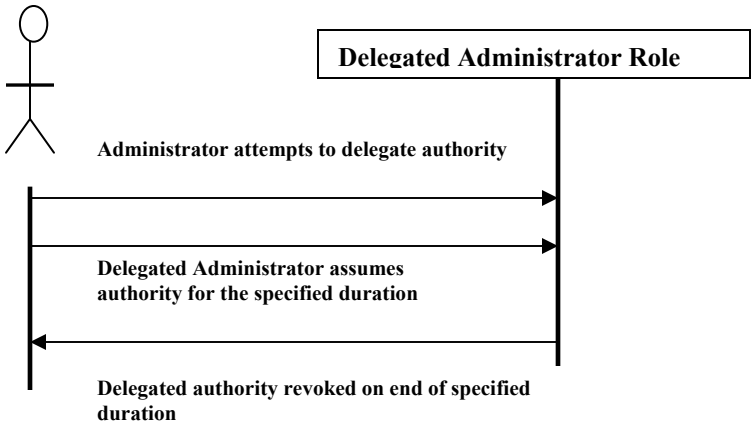


Table IV.17

Title	Delegate Administrator Role
Roles	Administrator Role, Delegated Administrator Role
Requirement#	
Pre-Conditions	An administrator should have been functioning.
Post-Conditions	Delegated Administrator temporarily assumes the Administrator authority.
Purpose	The purpose is to delegate authority temporarily in the absence of the Administrator.
Description	The Administrator decides who to delegate its authority temporarily and what to delegate. The Administrator also decides the duration of this delegation.



V References

1. http://csrc.nist.gov/publications/nistbul/csl95-12.txt
2. "A Role-based Use Case Model for Remote Data Acquisition Systems", Txomin Nieva, Alain Wegmann
3. "RBACManager: Implementing A Minimal Role Based Access Control Scheme ($RBAC_M$) Under the Windows NT 4.0 Workstation Operating System", W. Caelli, A. Rhodes
4. "Role-Based Access Control Models", Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman
5. "ROLES Project Final Report", http://groups.sims.berkeley.edu/doc-eng/projects/ROLES/roles-final-report.html
6. "Determining Role Rights from Use Cases", E. B. Fernandez, J. C. Hawkins
7. "Implement role based access control with attribute certificates", Wei Zhou, Christoph Meinel
8. "The Uses of Role Hierarchies in Access Control", Jonathan D. Moffet, Emil C. Lupu
9. "Role-based Authorization Constraints Specification Using Object Constraint Language", Gail-Joon Ahn, Michael E. Shin

10. “Addressing Repeatibility in Wireless Experiments using ORBIT Testbed”, Sachin Ganu, Haris Kremo, Richard Howard and Ivan Seskar.
11. “Orbit Testbed Software Architecture: Supporting Experiments as a Service”, Maximilian Ott, Ivan Seskar, Robert Siraccusa, Manpreet Singh
12. “Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols”, D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K Ramachandran, H. Kremo, R. Siracusa, H. Liu, M.Singh
13. “ORBIT Measurements Framework and Library (OML): Motivations, Design, Implementation, and Features”, Manpreet Singh, Maximilian Ott, Ivan Seskar