ISSUE TRACKING SYSTEM (ITS)

(Project V2 Report)

Revision History

Date	Version	Description	Author
03/15/2007	0.5	Draft	Project team
03/24/2007	1.0	Update Project Plan, Key Requirement	Project team
04/05/2007	1.5	Revise document according to feedback or Professors. Changes includes: - Project scope: explain more details of managing project profile, managing issues. - Key Requirement: adds non-functional requirement and updates use case model. - Business case: update cost & benefit analysis - Potential solution - Risk list: update high risk use cases - Project plan: update plan to attack high risk components first. - Business process model: change BPM and add analysis. - Glossary	Project team
04/06/2007	2.0	- Use case Detail - ER Diagram - DFD Diagram - Class Diagram - Package Diagram - Sequence Diagram	Project team
05/05/2007	3.0	- Refine Use case Detail - Refine Diagram - Add User guidelines - Add installation guidelines - Add Test plan - Add Test case	Project team

Table of Contents

1. Project vision

Nowadays, Vietnamese economy increases sharply, the software demand of many companies also raises considerably. Many applications that customers order Can Tho Software Park are bigger and more complex. Customer's quality requirements are higher and higher. As we know, project issue management will affect strongly future quality of software. However, present issue management of the company is offline and through project members. Project manager have to keep track of status of all project issues and assign project member to solve them. It is very difficult for project members to support together by giving comments or ideas to solve problems. Therefore, issue management of the company is too complicated and doesn't follow rational unify process (RUP). To overcome these challenges, we would like to build an online Issue Tracking System to keep track of issues tightly. With this system we can solve many problems as follow:

+Manage issue of projects follow RUP (keeping track of issues occur on each iteration of each phase)

+ Determine severity level of issues to solve basing on priority order.

+ Every project member is easy to get list of issues quickly that he/she has to solve and this list is up-to-date.

+ Project members can contribute their ideas to solve the issues.

+ Project manager can view issue chart to get a perspective about the project issue that he manage.

+ Manage and report of issues per build. According RUP process, we will have many builds of artifact corresponding to iterations. When someone submits an issue, he must attach information about the build number in which this issue occurs.

2. Project scope

The scope of this project can be described as follow:

- Issue tracking system allows each user manage issues of software project based on RUP architecture. It provides some convenient functions such as:
    + Manage project profile and assign project member to project. Managing project profile helps manager follow information of project. When a project profile was created, it has default information such as: the number of phases, iterations of each phase, artifacts or each iteration…Such information aims to follow the RUP process. Project manager can customize or scale according to the scope or nature of project. For example, the small project may have less iteration or artifact.
    + Manage right and role of each user in project.
    + Manage issues according to each phase and artifact. Managing issue help keep track information issue such as severity, priority, environment, build, etc. This aims to facilitate submitting and correcting issue. No issue will be lost or misunderstood by following the standard template.
    + Export issues to XML file.
    + Create report chart.
- We plan to build a web-based system which consists of two subsystems:
    + Administration System: manage user and project information.
    + Issue Tracking System: manage issue of a specific project from beginning to the end.

- Stakeholders who benefit from this system are: company administrator, project administrator, project manager, and normal user (designer, programmer, and tester).

Brief scope of our project

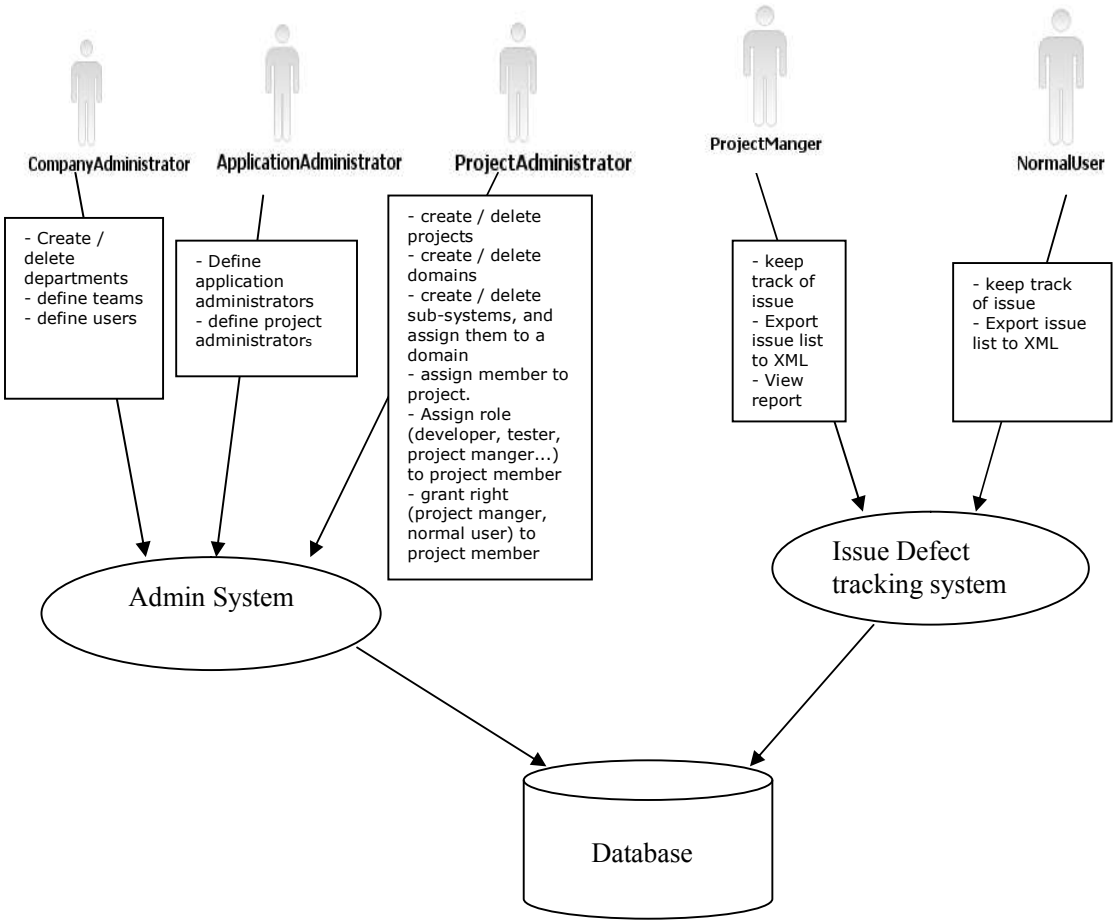


Figure 2.1- Brief scope project

3. Key Requirement

In this project, we apply two methods for determining requirements:

- Group interviews: We invite key members who have experience in developing software project and are representative for each role in project. They include project manager, software developer, and tester. Each role will have different expectation about new system. These expectations were recorded in meeting minutes enclosed in Appendices of this document.

- Direct observation: We observe the process of creating, editing, and closing an issue. Through that, we collect which issue content to record and how to facilitate the process.

3.1. Functional requirement

3.1.1. Use case model

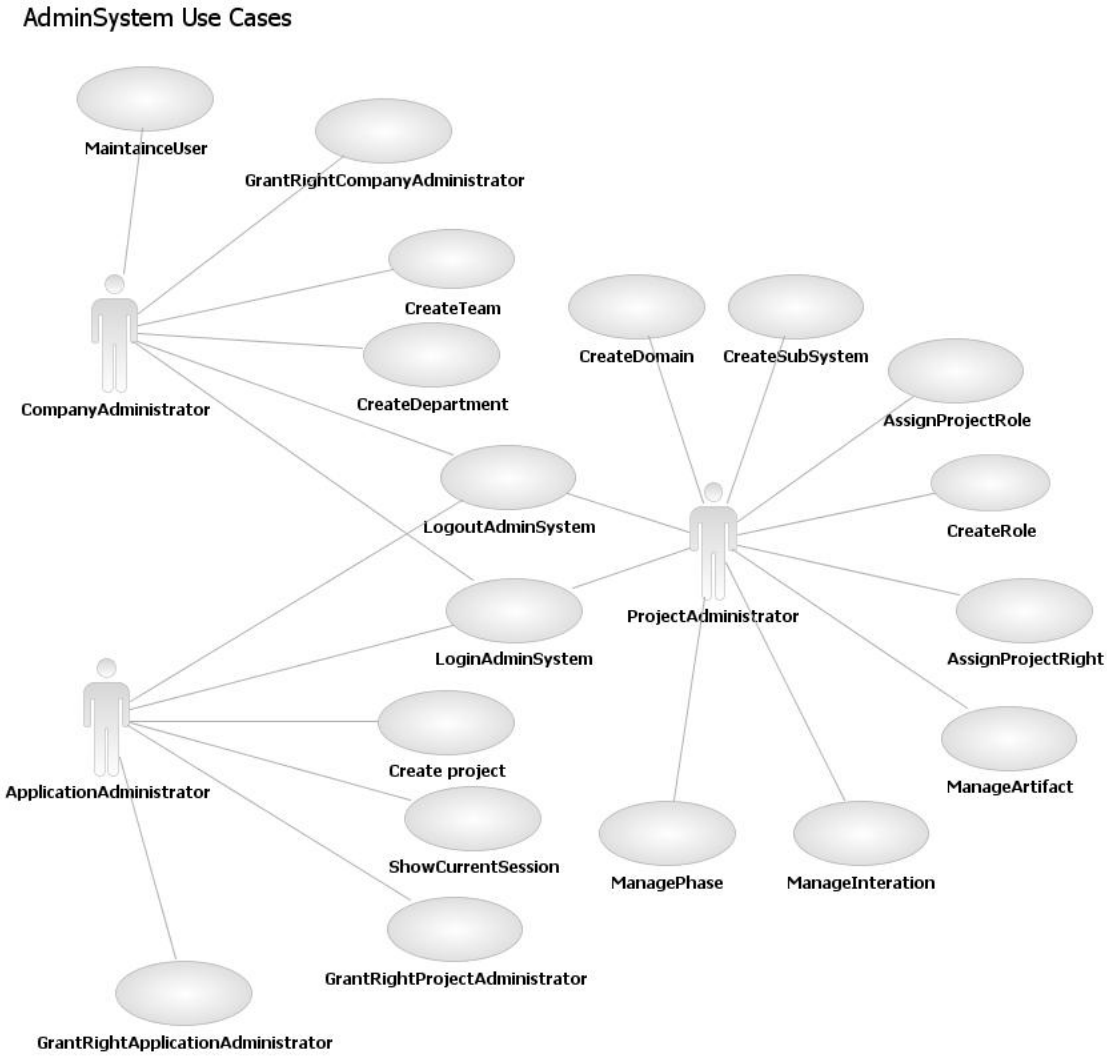


Figure 3.1- Admin system use case

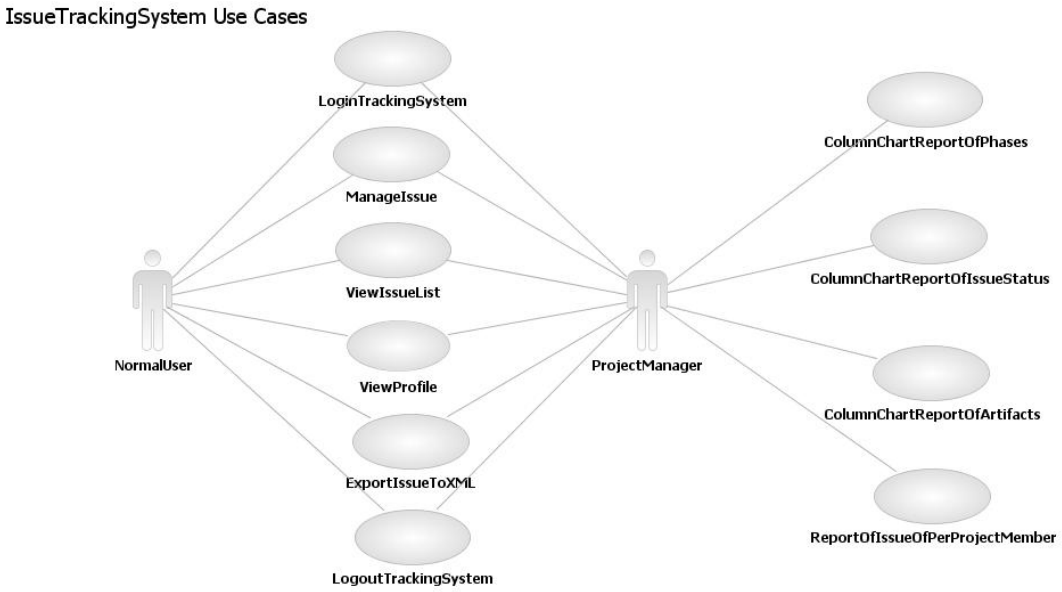


Figure 3.2- Issue tracking system use case.

3.1.2. Actor Semantics

Actors	Semantics
CompanyAdministrator (belong to Admin System)	Someone who has highest right in system. This person can manage departments, teams, and users of system.
ApplicationAdministrator (belong to Admin System)	A person who follow application's activities, define project and project's administrator.
ProjectAdministrator (belong to Admin System)	Someone who define project's information such as domain, subsystem, phase, iteration, artifact, and assign member to project.
NormalUser (belong to Defect Tracking System)	Someone who take part in project. He can view project issue information.
ProjectManager (belong to Defect Tracking System)	Someone who is responsible for managing his project.

Table: Actor semantics

3.1.3. Use case Semantics

Use cases	Semantics
MaintainUser	Company administrator can maintain users of system.
GrantRightCompanyAdministrator	Company administrator can grant company administrator right to any user in system.
CreateTeam	Company administrator can create, delete team
CreateDepartment	Company administrator can create, delete department
LoginAdminSystem	Describe activities relating to login into Admin system
LogoutAdminSystem	Describe activities relating to logout of Admin system
CreateProject	This use case describes rules of creating/deleting a project.
ShowCurrentSession	This use cases describe requirement of showing current session and history of system.
GrantRightProjectAdministrator	Application administrator can grant **project administrator** right to any user in system.
GrantRightApplicationAdministrator	Application administrator can grant **application administrator** right to any user in system.
CreateDomain	This use case describes rules of creating/deleting a domain.
CreateSubSystem	This use case describes rules of creating/deleting sub-systems for a specific domain of the project
AssignProjectRole	Project Administrator can assign a specific role to each project member.
CreateRole	Project Administrator can create roles for project. Each member in project will have a specific role (title).
AssignProjectRight	Project Administrator can assign a specific right to each project member.
ManagePhase	Project Administrator can define phases of his project
ManageIteration	Project Administrator can define iterations for each phase in project.
ManageArtifact	Project Administrator can define artifacts needed for each phase.
LoginTrackingSystem	Describe activities relating to login into Tracking system
ManageIssue	This use case describes rules of maintaining issues.
ViewIssueList	This use case describes activities to view list of issues
ViewProfile	This use case describes activities to view user's profile.
ExportIssueToXML	This use case describes exporting Issues to XML file.
LogoutTrackingSystem	Describe activities relating to logout of Tracking system
ColumnChartReportOfPhases	Report issues of each phase. This report is presented as column chart.
ColumnChartReportOfIssueStatus	Report issues according to status of issue. This report is presented as column chart.
ColumnChartReportOfArtifacts	Report issues according to a selected artifact. Show severity of issues. This report is presented as column

	chart.
ReportOfIssuePerProjectMember	Report issues of a specific project member.

3.1.4. Use case detail

Use case: **AddUser**
ID: **UC-01**
Brief description: This Use case is used to Add user of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with Company Administration right and click on tab "User".
Main flow: 1. The use case starts when the actor selects "Add' button. 2. The system displays: Textbox: User Name, Full Name, Password, Email. The default values of these fields are blank. Combo box: show list of departments Button: Save, Cancel. 3. Actor will enters correct information in all blank fields, selects department and clicks "Save" button. 4. System will verify information entered by user. If all information is correct, system will store new user information into database and show message "This user has been stored successfully in database. 5. Use case ends
Post conditions: 1. The system will turn back to user list screen with new user shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show a error message "The information entered is not correct, please check and try again."

Use case: **UpdateUser**
ID: **UC-02**
Brief description: This Use case is used to update user of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with Company Administration right and click on tab "User".
Main flow: 1. The use case starts when the actor checks on radio button corresponding to user needed to update and lick on Edit button. Or Actor can lick on the hyperlink username. 2. The system loads information of selected user and displays: Textbox: User Name, Full Name, Password, Email. The values of these fields are loaded of selected user information. Combo box: show list of departments Button: Save, Cancel. 3. Actor will enters correct information in all blank fields and click "Save" button.

4. System will verify information entered by user. If all information is correct, system will store new user information into database and show message “This user has been stored successfully in database. 5. Use case ends
Post conditions: 1. The system will turn back to user list screen with new user shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show a error message “The information entered is not correct, please check and try again.”

Use case: **DeleteUser**
ID: **UC-03**
Brief description: This use case is used to delete user of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with Company Administration right and click on tab “User” to show user list.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to user needed to delete and lick on Delete button. 2. System deletes the selected user from database and load new information into user table 3. Use case ends
Post conditions: 1. The system will turn back to user list screen with new user shown in list.
Alternative flows: None

Use case: **GrantRightCompanyAdministrator**
ID: **UC-04**
Brief description: The goal of this Use case show the actor can grant company administrator right to any user in system.
Primary actors: Company Administrator
Secondary actors: none
Preconditions: 1. Actor has successfully logged into system with Company Administration right.
Main flow: 1. The use case starts when the actor selects tab “Company administrator” to show list of user granted company right. 2. The system displays: Combo box: show list of users Button: Insert, Remove image on each user. Table: List of users granted company right

3. Actor selects user and clicks “Insert” button to grant company right to user or click ‘Remove image’ on each user to remove company right. 4. Use case ends.
Post conditions: 1. The system lists all users in a table to confirm.
Alternative flows: NA

Use case: **AddTeam**
ID: **UC-05**
Brief description: This Use case is used to Add Team of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with Company Administration right and click on tab “Team” to show list team.
Main flow: 1. The use case starts when the actor selects “Add” button. 2. The system displays: Textbox: Team Name. Button: Save. 3. Actor will enter correct information in blank field and clicks “Save” button. 4. System will verify information entered by user. If information is correct, system will store new team information into database and show message “This team has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to team list screen with new team shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **UpdateTeam**
ID: **UC-06**
Brief description: This Use case is used to update team of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with company administration right and click on tab “Team” to show list team.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to team needed to update and lick on “Edit” button. 2. The system loads information of selected team and displays: Textbox: Team Name. Button: Save. 3. Actor will enter correct information in all blank fields and click “Save” button. 4. System will verify information entered by actor. If information is correct, system will store update team information into database and show message “This team has been stored successfully

in database". 5. Use case ends
Post conditions: 1. The system will turn back to team list screen with updated user shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message "The information entered is not correct, please check and try again."

Use case: **DeleteTeam**
ID: **UC-07**
Brief description: This Use case is used to delete team of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: Actor has successfully logged into system with Company Administration right and click on tab Team to show team list.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to team needed to delete and lick on "Delete" button. 2. System deletes the selected team from database and load new information into team table 3. Use case ends
Post conditions: 1. The system will turn back to team list screen with updated team shown in list.
Alternative flows: None

Use case: **AddDepartment**
ID: **UC-08**
Brief description: This use case is used to add department of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with company administration right and click on tab "department" to show department list.

Main flow: 1. The use case starts when the actor selects add department button. 2. The system displays: Textbox: Department Name. Button: Save. 3. Actor will enter correct information in blank field and clicks “Save” button. 4. System will verify information entered by actor. If information is correct, system will store new department information into database and show message “This department has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to department list screen with new department shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **UpdateDepartment**
ID: **UC-09**
Brief description: This Use case is used to update department of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into system with company administration right and click on tab “Department” to show department list.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to department needed to update and lick on “Edit” button. 2. The system loads information of selected department and displays: Textbox: Department Name. Button: Save. 3. Actor will enter correct information in blank field and click “Save” button. 4. System will verify information entered by actor. If information is correct, system will store update department information into database and show message “This department has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to department list screen with updated department shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **DeleteDepartment**
ID: **UC-10**
Brief description: This use case is used to delete department of system
Primary actors: Company Administrator
Secondary actors: None
Preconditions: Actor has successfully logged into system with company administration right and click on tab “department” to show department list.

Main flow: 1. The use case starts when the actor checks on radio button corresponding to department needed to delete and lick on “Delete” button. 2. System deletes the selected department from database and load new information into department table. 3. Use case ends
Post conditions: 1. The system will turn back to department list screen with updated list shown in list.
Alternative flows: None

Use case: **LoginAdminSystem**
ID: **UC-11**
Brief description: Describe activities relating to login into admin system.
Primary actors: Company administrator, Project administrator
Secondary actors:
Preconditions: None
Main flow: 1. The use case starts when the actor want to login to the system. 2. The system displays the form Label: Username, Password. Button: Login 3. The actor input username and password. 4. The system will verify information entered by actor and allows the actor to login to the system 5. Use case ends.
Post conditions: 1. The system will show information on admin page to actor
Alternative flows: Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **LogoutAdminSystem**
ID: **UC-12**
Brief description: Describe activities relating to logout of Admin system
Primary actors: Company administrator, Project administrator.
Secondary actors:
Preconditions: None
Main flow: 1. This use case starts when the user select “Disconnect” 2. The system accepts and set the session of user is FALSE, then the system transfer to home screen 3. Use case ends.
Post conditions: 1. The user has been out of system

Alternative flows: None

Use case: **AddProject**
ID: **UC-13**
Brief description: This use case is used to add project of system
Primary actors: Company Administrator, Application Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into Application system with application right and click on tab "Project" to show list project.
Main flow: 1. The use case starts when the actor selects add project button. 2. The system displays: Textbox: Project Name, Project Description. Check box: Copy Template. Button: Save. 3. Actor will enter correct information in blank field, if actor want to make template for phase, iteration, artifact, issue type, issue status to project. Actor must check on check box "Copy Template" then clicks "Save" button. 4. System will verify information entered by actor. If information is correct, system will store new project information into database and show message "This project has been stored successfully in database". 5. Use case ends
Post conditions: 1. The system will turn back to project list screen with new project shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message "The information entered is not correct, please check and try again."

Use case: **UpdateProject**
ID: **UC-14**
Brief description: This use case is used to update project of system
Primary actors: Company Administrator, Application Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into application system with application right and click on tab "Project" to show list project.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to project needed to update and lick on "Edit" button. 2. The system loads information of selected Project and displays: Textbox: Project Name, Project Description. Button: Save. 3. Actor will enter correct information in all blank fields and click "Save" button. 4. System will verify information entered by actor. If information is correct, system will store update project information into database and show message "This project has been updated successfully in database". 5. Use case ends

Post conditions: 1. The system will turn back to project list screen with updated project shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **DeleteProject**
ID: **UC-15**
Brief description: This use case is used to delete project of system
Primary actors: Company Administrator, Application Administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into application system with application right and click on tab “Projects” to show list project.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to project needed to delete and lick on “Delete” button. 2. System deletes the selected project from database and load new information into project table. 3. Use case ends
Post conditions: 1. The system will turn back to project list screen with updated list shown in list.
Alternative flows: None

Use case: **ShowCurrentSession**
ID: **UC-16**
Brief description: This Use case describes current session and history of system
Primary actors: Application Administrators
Secondary actors:
Preconditions: 1. Actor has successfully logged into application administration field 2. Actor has ever accessed into system 3. The clock of window system must be exactly.
Main flow: 1. Actor observes all current sessions and history of system 2. Actor clicks “clear history” button to clear history if it is necessary 3. Actor clicks “clear all” button to clear both current session and history if it also is necessary to free memory. 4. Use case ends
Post conditions: 1. If actor clears all, the screen will space and the system will verify current session and history again.
Alternative flows:

Use case: **GrantRightApplicationAdministrator**
ID: **UC-17**

Brief description: This Use case shows the actor can grant application administrator right to any user in system.
Primary actors: Company Administrator, Application Administrator.
Secondary actors: none
1. Actor has successfully logged into application system with application right.
Main flow: 1. The use case starts when the actor selects tab “Application administrator”. 2. The system displays: Combo box: show list of Users Button: Insert, Remove image on each user. Table: List of users granted company right 3. Actor selects user and clicks “Insert” button to grant application right to user or click ‘Remove image’ on each user to remove application right. 4. Use case ends
Post conditions: 1. The system lists all users in a table to confirm.
Alternative flows: NA

Use case: **GrantRightProjectAdministrator**
ID: **UC-18**
Brief description: This use case shows the actor can grant project administrator right to any user in system.
Primary actors: Company Administrator, Application Administrator.
Secondary actors: none
1. Actor has successfully logged into application system with application right.
Main flow: 1. The use case starts when the actor selects tab “Project administrator.” 2. The system displays: Combo box: List of Users, list of Project. Button: Insert, Remove image on each user. Table: List of users granted project administrator. 3. Actor selects user and selects project, which user manage then clicks “Insert” button to grant project administrator right to user or click “Remove image” on each user to remove project administrator right. 4. Use case ends
Post conditions: 1. The system lists all users in a table to confirm.
Alternative flows: NA

Use case: **Add Domain**
ID: **UC-19**
Brief description: This use case is used to add project of system
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Domain” to show domain list.

Main flow: 1. The use case starts when the actor selects “Add” button. 2. The system displays: Textbox: Project Name. Button: Save. 3. Actor will enter correct information in blank field and clicks “Save” button. 4. System will verify information entered by actor. If information is correct, system will store new domain of project information into database and show message “This domain has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to domain list of project screen with new domain shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **UpdateDomain**
ID: **UC-20**
Brief description: This use case is used to update domain of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Domains” to show domain list.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to domain of project needed to update and lick on “Edit” button. 2. The system loads information of selected domain of project and displays: Textbox: Domain Name. Button: Save. 3. Actor will enter correct information in all blank fields and click “Save” button. 4. System will verify information entered by actor. If information is correct, system will store update domain of project information into database and show message “This domain has been updated successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to domain of project list screen with updated domain shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **DeleteDomain**
ID: **UC-21**
Brief description: This use case is used to delete domain of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab ‘Domains” to show domain list

Main flow: 1. The use case starts when the actor checks on radio button corresponding to domain of project needed to delete and lick on “Delete” button. 2. System deletes the selected domain of project from database and load new information into domain of project table. 3. Use case ends
Post conditions: 1. The system will turn back to domain list screen with updated list shown in list.
Alternative flows: None

Use case: **AddSubSystem**
ID: **UC-22**
Brief description: This Use case is used to add subsystem of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab Subsystem. 2. Actor has selected project and domain of project.
Main flow: 1. The use case starts when the actor selects “Add” button. 2. The system displays: Textbox: Subsytem Name. Button: Save. 3. Actor will enter correct information in blank field and clicks “Save” button. 4. System will verify information entered by actor. If information is correct, system will store new subsystem of project information into database and show message “This subsystem has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to subsystem list of project screen with new subsystem shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **UpdateSubsystem**
ID: **UC-23**
Brief description: This Use case is used to update subsystem of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab Subsystem. 2. Actor has selected project and domain of project.

Main flow: 1. The use case starts when the actor checks on radio button corresponding to subsystem of project needed to update and lick on “Edit” button. 2. The system loads information of selected subsystem of project and displays: Textbox: Subsystem Name. Button: Save. 3. Actor will enter correct information in blank fields and click “Save” button. 4. System will verify information entered by actor. If information is correct, system will store updated subsystem of project information into database and show message “This subsystem has been updated successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to subsytem of project list screen with updated subsystem shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **DeleteSubsystem**
ID: **UC-24**
Brief description: This use case is used to delete Subsystem based on domain of project
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab Domain. 2. Actor has selected project and domain of project
Main flow: 1. The use case starts when the actor checks on radio button corresponding to domain of project needed to delete and lick on “Delete” button. 2. System deletes the selected domain of project from database and load new information into domain of project table. 3. Use case ends
Post conditions: 1. The system will turn back to domain list screen with updated list shown in list.
Alternative flows: None

Use case: **AddRole**
ID: **UC-25**
Brief description: This Use case is used to roles of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab Roles. 2. Actor has selected project.

Main flow: 1. The use case starts when the actor selects “Add” button. 2. The system displays: Textbox: Role Name. Button: Save. 3. Actor will enter correct information in blank field and clicks “Save” button. 4. System will verify information entered by actor. If information is correct, system will store new role of project information into database and show message “This role has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to role list of project screen with new role shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **UpdateRole**
ID: **UC-26**
Brief description: This Use case is used to update role of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Roles”. 2. Actor has selected project.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to role of project needed to update and lick on “Edit” button. 2. The system loads information of selected role of project and displays: Textbox: Role Name. Button: Save. 3. Actor will enter correct information in blank fields and click “Save” button. 4. System will verify information entered by actor. If information is correct, system will store updated role of project information into database and show message “This role has been updated successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to role of project list screen with updated role shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **DeleteRole**
ID: **UC-27**
Brief description: This use case is used to delete role of project
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Role”.

2. Actor has selected project.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to Role of project needed to delete and lick on “Delete” button. 2. System deletes the selected role of project from database and load new information into role of project table. 3. Use case ends
Post conditions: 1. The system will turn back to role list screen with updated role list shown in list.
Alternative flows: None

Use case: **AssignProjectMembers**
ID: **UC-28**
Brief description: This use case shows the actor can grant project right and role to any user in system.
Primary actors: Company Administrator, Application Administrator.
Secondary actors: none
1. Actor has successfully logged into project administration system with project administration right.
Main flow: 1. The use case starts when the actor selects tab “Project members.” 2. The system displays: Combo box: List of Projects, list of Users, list of Roles, list of Right Button: Insert, Remove image on each user. Table: List of users granted project member. 3. Actor selects user, selects project, select role and select right , which to be assigned to user then clicks “Insert” button or click “Remove image” on each user to remove project members 4. Use case ends
Post conditions: 1. The system lists all users in a table to confirm.
Alternative flows: NA

Use case: **AddPhase**
ID: **UC-29**
Brief description: This Use case is used to add phase of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Phase”. 2. Actor has selected project.

Main flow: 1. The use case starts when the actor selects “Add” button. 2. The system displays: Textbox: Phase Name. Button: Save. 3. Actor will enter correct information in blank field and clicks “Save” button. 4. System will verify information entered by actor. If information is correct, system will store new phase of project information into database and show message “This phase has been stored successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to phase list of project screen with new phase shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **UpdatePhase**
ID: **UC-30**
Brief description: This use case is used to update phase of project.
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Phase”. 2. Actor has selected project.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to phase of project needed to update and lick on “Edit” button. 2. The system loads information of selected phase of project and displays: Textbox: Phase Name. Button: Save. 3. Actor will enter correct information in blank fields and click “Save” button. 4. System will verify information entered by actor. If information is correct, system will store updated phase of project information into database and show message “This phase has been updated successfully in database”. 5. Use case ends
Post conditions: 1. The system will turn back to phase of project list screen with updated phase shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **DeletePhase**
ID: **UC-31**
Brief description: This use case is used to delete phase of project
Primary actors: Project administrator
Secondary actors: None
Preconditions: 1. Actor has successfully logged into project administration system with project administration right and click on tab “Phase”.

2. Actor has selected project.
Main flow: 1. The use case starts when the actor checks on radio button corresponding to phase of project needed to delete and lick on “Delete” button. 2. System deletes the selected phase of project from database and load new information into phase of project table. 3. Use case ends
Post conditions: 1. The system will turn back to phase list screen with updated phase list shown in list.
Alternative flows: None

Use case: **ManageIteration**
ID: **UC-32**
Brief description: This Use case show the actor can define iterations for each phase in project
Primary actors: Project Administrators
Secondary actors:
Preconditions: 1. The actor is logged into the system with project administrator right
Main flow: 1. This use case starts when the actor select tab “Iterations” 2. The system displays Combo box: List of projects, list of phases. Button: Add, Remove, Update. Table: List of iterations of project. 3. The actor selects project, phase and choice “New” button to add iteration or checks on radio button corresponding to iteration of project needed to delete/update 4.1 If the actor choice “New” button. 4.1.1 The system displays: Textbox: Iteration name. Button: Save. 4.1.2 Actor will enter correct information in blank field and clicks “Save” button. 4.1.3 System will verify information entered by actor. If information is correct, system will store new iteration of project information into database and show message “This iteration has been stored successfully in database”. 4.2 If the actor choice “Update” button. 4.2.1. The system loads information of selected iteration of project and displays: Textbox: Iteration Name. Button: Save. 4.2.2 Actor will enter correct information in blank fields and click “Save” button. 4.2.3 System will verify information entered by actor. If information is correct, system will store updated iteration of project information into database and show message “This iteration has been updated successfully in database”. 4.3 If the actor choice “Delete” button. 4.3.1. System deletes the selected iteration of project from database and load new information into iteration of project table. 5. Use case ends

Post conditions: The system will turn back to iteration of project list screen with updated iteration shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **ManageArtifact**
ID: **UC-33**
Brief description: Project Administrator can define artifacts needed for each phase.
Primary actors: Project manager
Secondary actors:
Preconditions: 1. The actor is logged into the system.
Main flow: 1. This use case starts when the actor select tab “Artifacts” 2. The system displays Combo box: List of projects, list of phases, list of iterations. Button: Add, Remove, Update. 3. The actor selects project, phase, iteration and choice “New” button to add artifact or checks on radio button corresponding to artfact of project needed to delete/update 4.1 If the actor choice “New” button. 4.1.1 The system displays: Textbox: Artifact name. Button: Save. 4.1.2 Actor will enter correct information in blank field and clicks “Save” button. 4.1.3 System will verify information entered by actor. If information is correct, system will store new artifact of project information into database and show message “This artifact has been stored successfully in database”. 4.2 If the actor choice “Update” button. 4.2.1. The system loads information of selected artifact of project and displays: Textbox: Artifact Name. Button: Save. 4.2.2 Actor will enter correct information in blank fields and click “Save” button. 4.2.3 System will verify information entered by actor. If information is correct, system will store updated artifact of project information into database and show message “This artifact has been updated successfully in database”. 4.3 If the actor choice “Delete” button. 4.3.1. System deletes the selected artifact of project from database and load new information into artifact of project table. 5. Use case ends
Post conditions: The system will turn back to artifact of project list screen with updated artifact shown in list.
Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message “The information entered is not correct, please check and try again.”

Use case: **LoginTrackingSystem**
ID: **UC-34**
Brief description: Describe activities relating to login into Tracking system.
Primary actors: Project manager, normal user

Secondary actors:
Preconditions: None
Main flow: 1. The use case starts when the actor want to login to the system. 2. The system displays the form Label: Username, Password. Button: Login 3. The actor input username and password. 4. The system will verify information entered by actor and allows the actor to login to the system 5. Use case ends.
Post conditions: 1. The system will show information on admin page to actor
Alternative flows: Alternative flows: At step 4 in main flow, If system check that information is not correct, system will show error message "The information entered is not correct, please check and try again."

Use case: **ManageIssue**
ID: **UC-35**
Brief description: This use case describes rules of maintaining issues.
Primary actors: Project manager, normal user
Secondary actors:
Preconditions: 1. The actor is logged into the system
Main flow: 1. The use case starts when the actor selects Issue tab. 2. The system displays: • "Status" combo box: Open, Fixed, Pending, Closed • "Severity" combo box: Minor, Medium, High, Fatal • "Assigned to" combo box: list of all project members • "Created by" combo box: list of all project members • "Type" combo box: Bug, change request, evolution, suggested • "Priority" combo box: Low, Medium, High • "Phase" combo box: list of all phases of project • "Iteration" combo box: list of all iterations of the selected phase above • "Artifact" combo box: list of all artifacts of the selected iteration above • "Search" button: • A table list of issues satisfying above criteria. The columns includes: Issue ID, title, severity, priority, status, Assigned to, Created by, Fixed Date. • Three buttons: New, Edit, and Delete. 3. The actor select issue and choice button 3.1 If the actor choice New button. 3.1.1 The system display a form for actor to add new information with fields: Title, Statue, Type of artifact, Project, Phase, Domain, Subsystem, artifact, build number. 3.1.2 The actor input information then choice Save button. 3.1.3 System will verify correctness of submitted data. If all information is correct, system save new issue into database and turn back to Issue tab. 3.2 If the actor choice Edit button. 3.2.1 The system display a form filled with information of the selected issue

3.2.2 The actor update and click Save button
3.2.3 System will verify correctness of submitted data. If all information is correct, system update issue into database and turn back to Issue tab.
3.3 If the actor choice Delete button.
3.3.1 The system asks for confirm "Are you sure to delete this issue", If actor agrees by clicking on Ok button, new information of issue will be updated into database and system turn back to Issue tab. If actor clicks on Cancel button, there's no thing change in database.
4. Use case ends.

Post conditions:
New information was updated into database

Alternative flows:
On main flow 3.1.3 and 3.2.3
1. If actor entered incorrect information. The system will notice errors and remind actor to correct information.
2. The actor enters information or clicks "Cancel" button to return main flow. The use case is end.

Use case: **ViewIssueList**
ID: **UC-36**
Brief description: This use case describes activities to view list of issues
Primary actors: Project manager, normal user.
Secondary actors:
Preconditions: 1. The actor is logged into the system
Main flow: 1. This use case starts when the actor select "Issue" tab. 2. The system displays: • "Status" combo box: Open, Fixed, Pending, Closed • "Severity" combo box: Minor, Medium, High, Fatal • "Assigned to" combo box: list of all project members • "Created by" combo box: list of all project members • "Type" combo box: Bug, change request, evolution, suggested • "Priority" combo box: Low, Medium, High • "Phase" combo box: list of all phases of project • "Iteration" combo box: list of all iterations of the selected phase above • "Artifact" combo box: list of all artifacts of the selected iteration above • "Search" button: • A table list of issues satisfying above criteria. The columns includes: Issue ID, title, severity, priority, status, Assigned to, Created by, Fixed Date. • Three buttons: New, Edit, and Delete. 3. Actor selects the wanted issue on the list to view detail.
Post conditions: None
Alternative flows: None

Use case: **ViewProfile**
ID: **UC-37**
Brief description: This use case describes activities to view user's profile

Primary actors: Company administrator, project administrator, project manager, normal user
Secondary actors:
Preconditions: 1. The actor is logged into the system
Main flow: 1. This use case starts when the actor select “Profile” 2. The system displays list of information, that related with actor such as: Your profile by project Domain, which actor belongs to Member of the domain Project domain and subsystem Member of all domains, profiles Change password 3. The actor selects one of these information. 4. The system accept, and display in detail
Post conditions: 1. None
Alternative flows: If the actor choice “Change password”. 1. The system requires the actor input new password. 2. Then the system accepts and store into database. Use case is end.

Use case: **ExportIssueToXML**
ID: **UC-38**
Brief description: This use case describes exporting Issues to XML file
Primary actors: Project manager, normal user.
Secondary actors:
Preconditions: 1. The actor is logged into the system
Main flow: 1. This use case starts when the actor select “Export to XML” 2. The system ask the actor mark the list of issues, that belongs to actor 3. The actor select issue the choice “Export” button 4. The system accept and display information on web browser by XML file 5. Use case ends.
Post conditions: 1. A new XML file has been created for the actor
Alternative flows: None

Use case: **LogoutTrackingSystem**
ID: **UC-39**
Brief description: Describe activities relating to logout of Tracking system
Primary actors: Project manager and Normal user
Secondary actors:
Preconditions: None

Main flow: 1. This use case starts when the user select “Log out” 2. The system accepts and set the session of user is FALSE, then the system transfer to home screen. 3. Use case ends.
Post conditions: 1. The user has been out of system
Alternative flows: None

Use case: **ColumnChartReportOfPhases**
ID: **UC-40**
Brief description: Report issues of each phase. This report is presented as column chart.
Primary actors:
Secondary actors:
Preconditions: 1. The actor is logged into the system.
Main flow: 1. This use case starts when the project manager select “Column Chart Report of Phases”. 2. The system displays a form to require user to input information of Phases such as: Type of phase, project, project member, date… 3. The user input information then choice “Report” button. 4. The system report column chart of phases. 5. Use case ends.
Post conditions: 1. The system displayed column chart report of artifact
Alternative flows: 1. When actor choice Report button but lack requirement information. The system will display error and require actor choose again. 2. The actor can return main flow or cancel. The use case will be end.

Use case: **ColumnChartReportOfIssueStatus**
ID: **UC-41**
Brief description: The Project Manager reports issues according to status of issue. This report is presented as column chart.
Primary actors: Project Manager
Secondary actors: None
Preconditions: 1. The actor is logged into the system
Main flow: 1. This use case starts when the project manager select “Column Chart Report of Issue Status”. 2. The system displays a form to require user to input information of issue such as: Type of issue, project what issue belongs to, and date. 3. The user select information then choice “Report” button. 4. The system report column chart of issue status 5. Use case ends.
Post conditions: 1. The system displayed column chart report of Issue statue

Alternative flows: 1. When actor choice Report button but lack requirement information. The system will display error and require actor choose again. 2. The actor can return main flow or cancel. The use case will be end.

Use case: **ColumnChartReportOfArtifacts**
ID: **UC-42**
Brief description: The project manager create column chart report according to selected artifact
Primary actors: Project Manager
Secondary actors:
Preconditions: 1. The actor is logged in to the system
Main flow: 1. This use case starts when the project manager select “Column Chart Report of Artifacts” 2. The system display a form to require user to input information such as: Type of artifact, project, date 3. The user input information then choice “Report” button 4. The system report column chart of artifact 5. Use case ends.
Post conditions: 1. The system displayed Column Chart Report of Artifact
Alternative flows: 1. When actor choice Report button but lack requirement information. The system will display error and require actor choose again. 2. The actor can return main flow or cancel. The use case will be end.

Use case: **ReportOfIssuePerProjectMember**
ID: **UC-43**
Brief description: Report issues of a specific project member.
Primary actors: Project Manager
Secondary actors: None
Preconditions: 1. The project manager is logged on to the Issue Tracking System
Main flow: 1. This use case starts when the project manager select “Report issue” 2. The system display a form to require user to input User ID, Projects, Joined Date 3. The user input information then choice “Report” button 4. The system report issues of specific project member following: project, phase, domain, subsystem, and artifact. 5. Use case ends.
Post conditions: 1. The system report issues of specific project member.
Alternative flows: 1. When actor choice Report button but did not select User ID. The system will display error and require choose again. 2. The actor can return main flow or cancel. The use case will be end.

3.2. Non-functional requirement

The performance of system must be acceptable in normal load. The responding time of system must not slower than five seconds except for reports (ten seconds).

4. Potential Solution

Through discussion, we specify one potential solution. This system will consist of two sub-systems:

- Administration system: manage activities relating to user and general information of project.
- Issue tracking system: mange issues and reports.

We will apply three tier layers model:

- Presentation: The Presentation layer is totally managed by Tomcat web sever. The main package that resides on this layer is JSP pages. Client uses web browser to access these JSP pages. For this layer, we will have some pages for Project Admin to enter initial information of a project such as title, description, phases, iterations, artifacts, domains, subsystems. We also create some pages for project members to enter issue profile, add comments for issue. Project manager can get some reports about issues in chart.
- Business layer: The Business layer consists of several server-side classes which are responsible for performing business logic. For this layer, we will have three main java beans: Project, Issue and project member java beans. Project java bean manages profiles of project. Project members manage information of members of project, including right and role of them in project. Issue java bean manages issue profile and makes some reports about issues of project.
- Data layer: This layer includes data access classes. These classes provide methods of accessing and updating data between the business layer and database. We use JDBC to connect to MySQL server. We will implement DBManager java bean to facilitate developer in implementing code. There will be some mapping classes such as IssueDB, ProjectDB, ProjectmeberDB to retrieve data from some tables of MySQL server.

User can access the system through Internet or LAN.

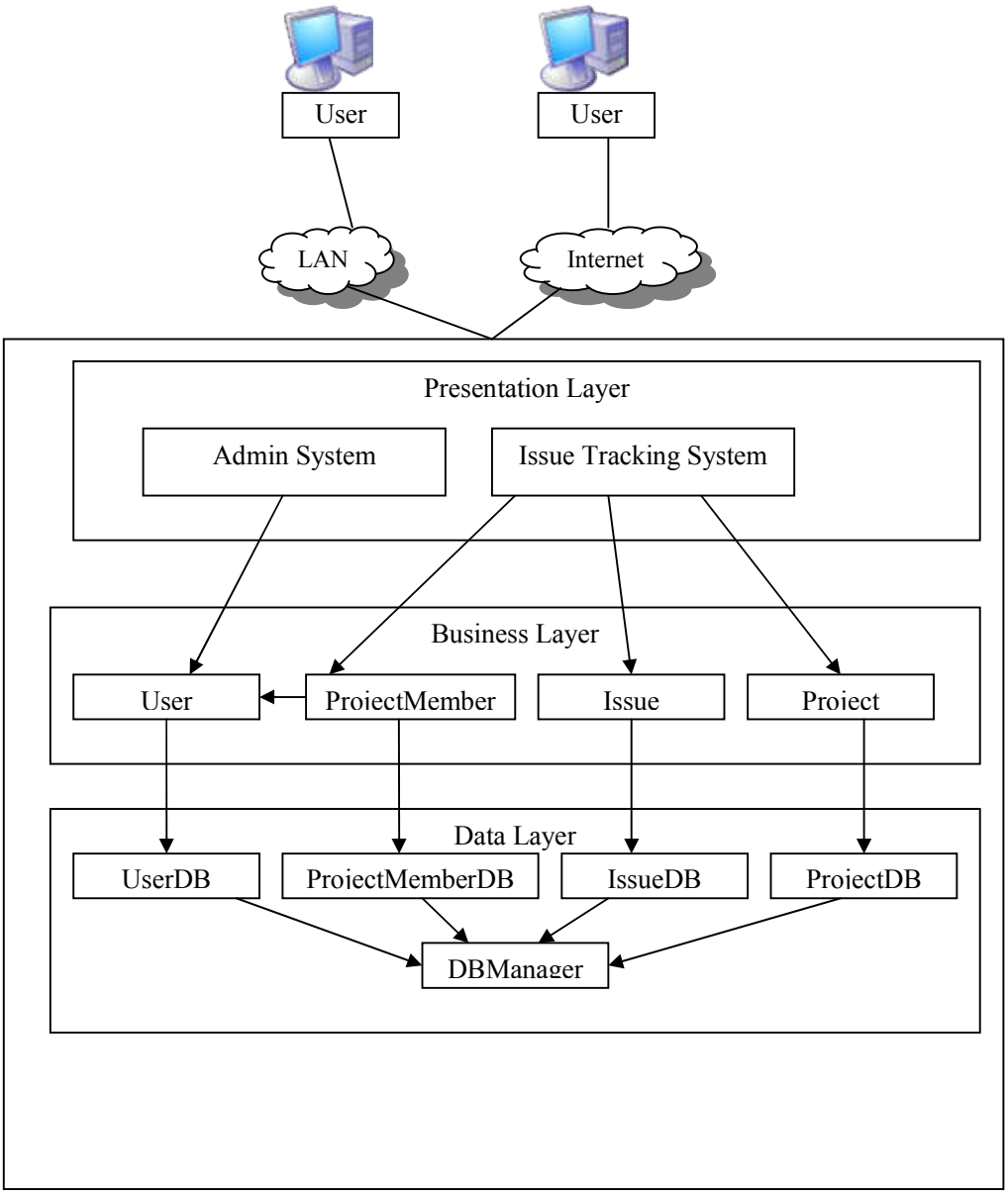


Figure 4.1- System architecture

5. Business Case

5.1. Cost

a. Hardware Cost

No.	Item	Quantity	User	Description	Cost	Total
1	Server	2		Use for database server and web server	2000$	4000$
2	Workstation	10	Company administrator, Project administrator, Project Manager, Normal user	Publish project, issue defect tracking	Existing	
	Total					4000$

b. Development cost

No	Item	Cost	Total
1	Development cost	3000$	3000$
2	User training	500$	500$
	Total		3500$

c. Recurring cost

Yearly maintain system: 100$

5.2. Benefit

a. Yearly tangible benefit

Table below displays analysis results of cost and turnover of company's projects in year 2006.

No.	Project	Time(month)	Cost(Salary for employees, Project Management, wasted properties)	Turnover
1	School management software for Cantho Economic and Technology College	8	2000	3000
2	Human resource and salary management for quantrung company	5	1800	2300
3	Hospital Management for Cantho 121 Hospital	8	2200	3000
4	Website school management Phan Ngoc Hien shool	5	2000	2500
5	Website for Cantho International Exhibit	5	2000	2500

	Farm			
6	Financial Management for Lehieu company	5	2000	2500
7	Hotel and Restaurant Management for Cantho Gold Hotel	6	2300	2700
8	Website for Department of Cantho Education and Training	7	2400	3000
9	Human resource and salary management for Department of Cantho Education and Training	8	2500	3000
10	Website for Mekongfish Company	4	2000	2500
11	Website for Hatien2 Cantho cement	4	2000	2500
12	Website for Vinhlong Department of Science and Technology	6	2400	3000
13	Demographic Management for Cantho Police Deparment	8	3300	4000
14	E-Learning Portal for cantho party committee	8	2700	3500
Total			31600	40000

Analysis information for year 2006:

Average cost per Project: 31600/14 = 2257.1429 $

Average turnover per each project = 2857.143 $

CSP made Plan for year 2007

+ Software and website: 20 Projects.

We had a discussion with financial and software department of CSP. The analysis table bellow is just an estimation of cost **for one project** which CSP may acquire if apply this system.

Without ITS				ITS applied			
Cost				Cost			
Analysis and Design	Construction	Testing	Project management	Analysis and Design	Construction	Testing	Project management
500	1000	500	250	450	950	400	220
Total: 500+1000+500+250= 2250				Total: 450+950+400+220= 2020			

Total cost without ITS = 20*2250= 45000 $
Total cost with ITS applied = 20*2020= 40400 $

We estimate the benefits that CSP will get:
- Error reduction.
- Improvement project management.
- Increase the quality value of product.
- Overall revenue of selling this product to other companies.

Reduction cost: 4600 $

b. Intangible benefit
- Increase company prestige by improving quality.
- Standardize software process.
- Faster decision making by giving detail reports of projects.

Estimated ROI:

	Year 0	Year 1	Year 2	Year 3	Total
Net economic benefit	0	4600	4600	4600	
Discount rate(8%)		0.925926	0.857339	0.793832	
PV Benefit	0	4259.259	3943.759	3651.628	
NPV of all BENEFITS	0	4259.259	8203.018	11854.65	11854.65
On-time cost	7500				
Recurring Cost		100	100	100	
Discount rate (8%)		0.925926	0.857339	0.793832	
PV of Recurring Costs		92.59259	85.73388	79.38322	
NPV COSTS	7500	7592.593	7678.326	7757.71	7757.71
Overall NPV					4096.936
Overall ROI					0.528112
Break-even Analysis					
Yearly NPV Cash Flow	0	4166.667	3858.025	3572.245	
Overall NPV Cash Flow	-7500	-3333.33	524.6914	4096.936	

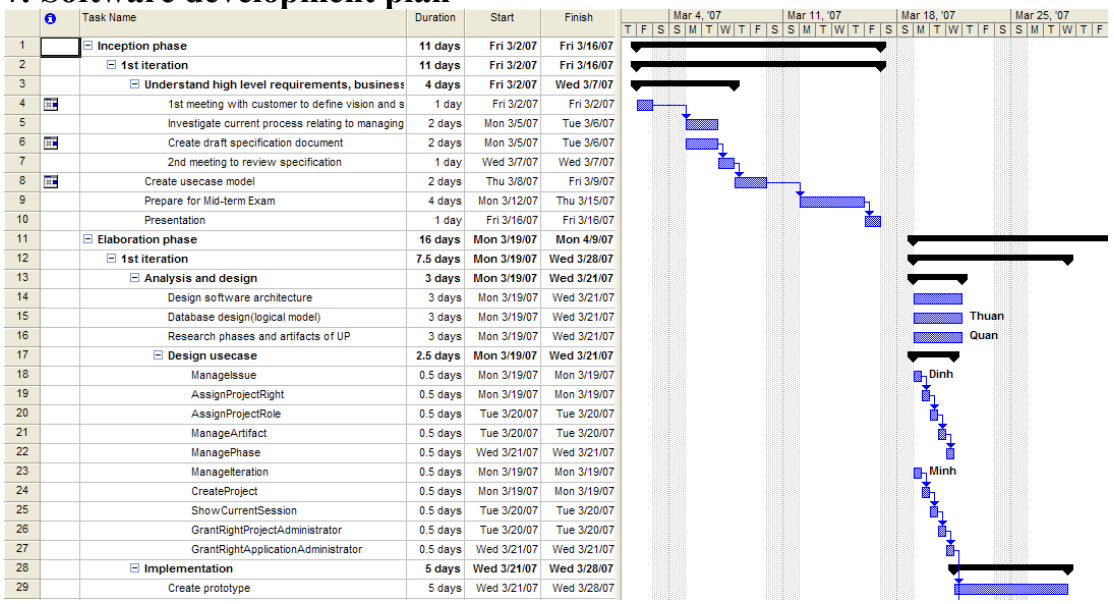
6. Risk List

The list below is the risks ordered by severity:

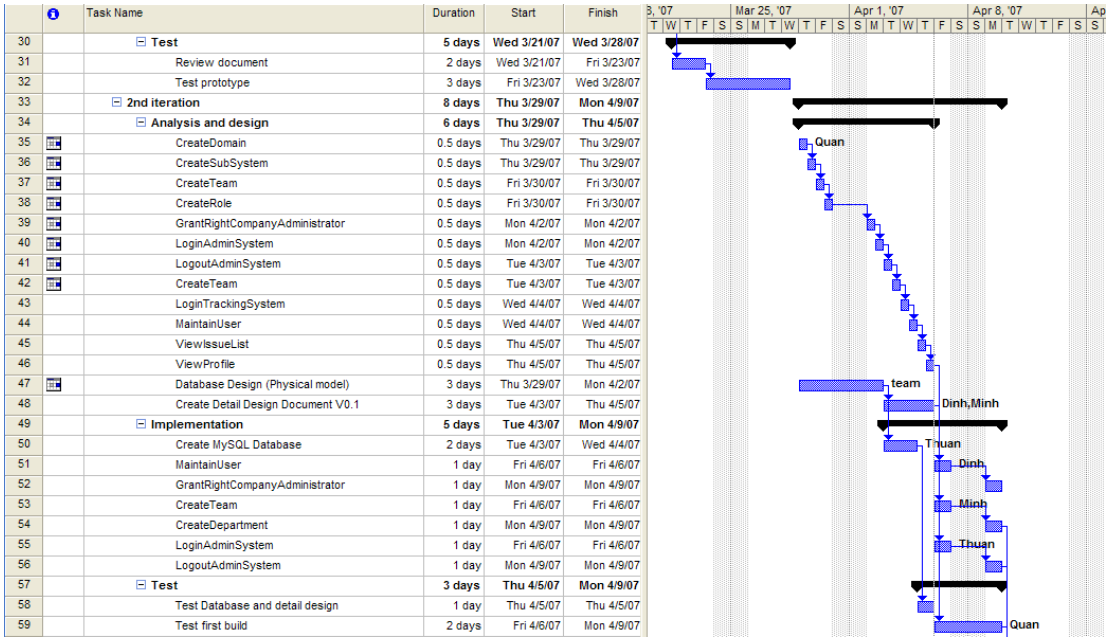
Risk	Description	Severity	Solution
Use case “Manage Issue”	This use case is complex and relates to lots of concept hard to understand.	High	Have a plan to design this use case as soon as possible.
Use cases relating to project such as “CreateProject”, “AssignProjectRight”, “AssignProjectRole”, “Manage Artifact”, ”ManagePhase”, ”ManageIteration”	These use cases are essential for understanding the whole project because they relate to concept of RUP process.	High	Attack these use cases soon to support mutual understanding among stakeholders.
Misunderstanding requirement	Lacking of time for directly discussion, far distance from customer site.	Average	Use other alternative ways to discuss with customer such as chat, e-mail, and phone to

			understand clearly requirement.
Not familiar with RUP	Don't know which artifacts needed to produce for each phase in RUP	Average	Research RUP early.

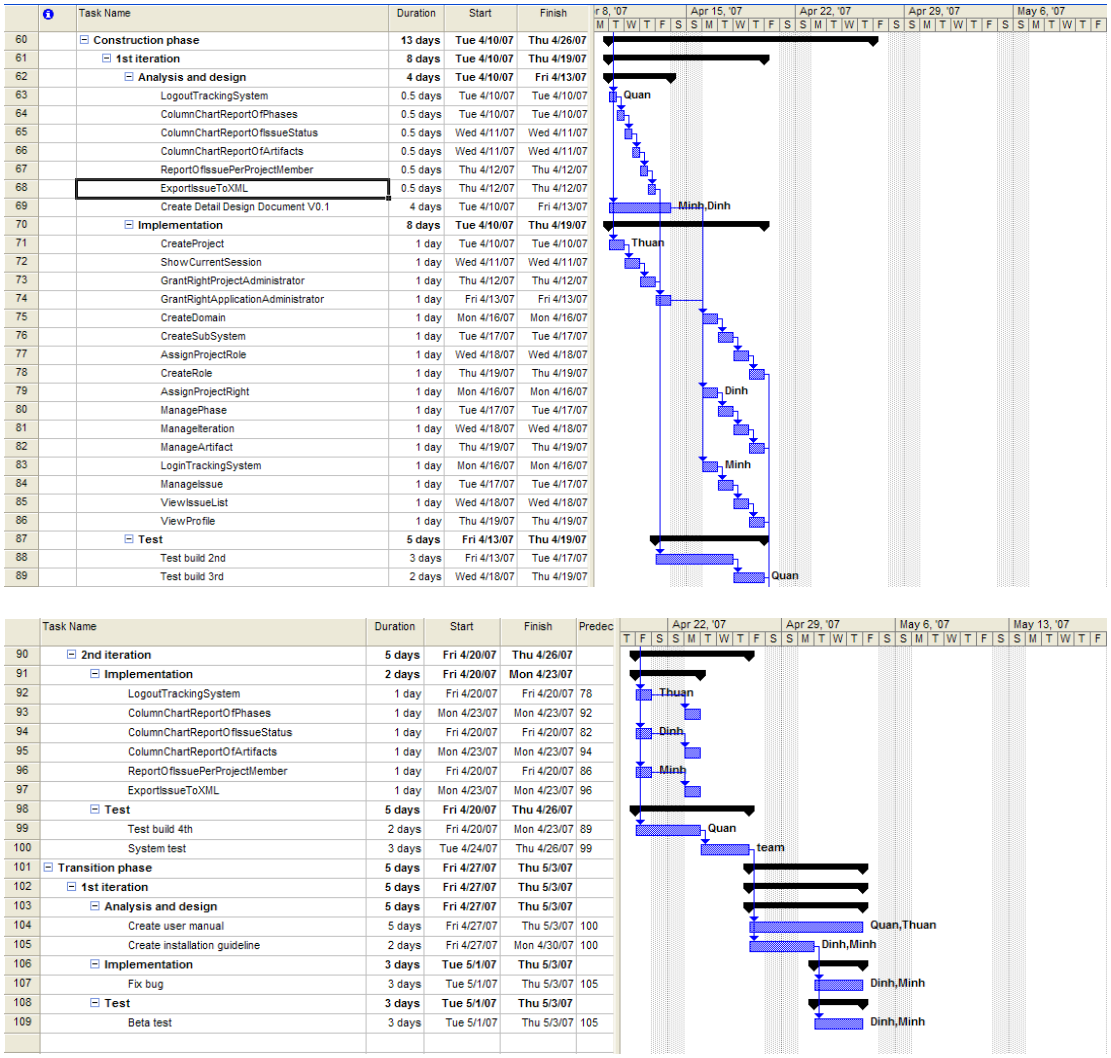
7. Software development plan



	Task Name	Duration	Start	Finish
1	**Inception phase**	**11 days**	**Fri 3/2/07**	**Fri 3/16/07**
2	**1st iteration**	**11 days**	**Fri 3/2/07**	**Fri 3/16/07**
3	**Understand high level requirements, business**	**4 days**	**Fri 3/2/07**	**Wed 3/7/07**
4	1st meeting with customer to define vision and s	1 day	Fri 3/2/07	Fri 3/2/07
5	Investigate current process relating to managing	2 days	Mon 3/5/07	Tue 3/6/07
6	Create draft specification document	2 days	Mon 3/5/07	Tue 3/6/07
7	2nd meeting to review specification	1 day	Wed 3/7/07	Wed 3/7/07
8	Create usecase model	2 days	Thu 3/8/07	Fri 3/9/07
9	Prepare for Mid-term Exam	4 days	Mon 3/12/07	Thu 3/15/07
10	Presentation	1 day	Fri 3/16/07	Fri 3/16/07
11	**Elaboration phase**	**16 days**	**Mon 3/19/07**	**Mon 4/9/07**
12	**1st iteration**	**7.5 days**	**Mon 3/19/07**	**Wed 3/28/07**
13	**Analysis and design**	**3 days**	**Mon 3/19/07**	**Wed 3/21/07**
14	Design software architecture	3 days	Mon 3/19/07	Wed 3/21/07
15	Database design(logical model)	3 days	Mon 3/19/07	Wed 3/21/07
16	Research phases and artifacts of UP	3 days	Mon 3/19/07	Wed 3/21/07
17	**Design usecase**	**2.5 days**	**Mon 3/19/07**	**Wed 3/21/07**
18	ManageIssue	0.5 days	Mon 3/19/07	Mon 3/19/07
19	AssignProjectRight	0.5 days	Mon 3/19/07	Mon 3/19/07
20	AssignProjectRole	0.5 days	Tue 3/20/07	Tue 3/20/07
21	ManageArtifact	0.5 days	Tue 3/20/07	Tue 3/20/07
22	ManagePhase	0.5 days	Wed 3/21/07	Wed 3/21/07
23	ManageIteration	0.5 days	Mon 3/19/07	Mon 3/19/07
24	CreateProject	0.5 days	Mon 3/19/07	Mon 3/19/07
25	ShowCurrentSession	0.5 days	Tue 3/20/07	Tue 3/20/07
26	GrantRightProjectAdministrator	0.5 days	Tue 3/20/07	Tue 3/20/07
27	GrantRightApplicationAdministrator	0.5 days	Wed 3/21/07	Wed 3/21/07
28	**Implementation**	**5 days**	**Wed 3/21/07**	**Wed 3/28/07**
29	Create prototype	5 days	Wed 3/21/07	Wed 3/28/07



	Task Name	Duration	Start	Finish
30	**Test**	**5 days**	**Wed 3/21/07**	**Wed 3/28/07**
31	Review document	2 days	Wed 3/21/07	Fri 3/23/07
32	Test prototype	3 days	Fri 3/23/07	Wed 3/28/07
33	**2nd iteration**	**8 days**	**Thu 3/29/07**	**Mon 4/9/07**
34	**Analysis and design**	**6 days**	**Thu 3/29/07**	**Thu 4/5/07**
35	CreateDomain	0.5 days	Thu 3/29/07	Thu 3/29/07
36	CreateSubSystem	0.5 days	Thu 3/29/07	Thu 3/29/07
37	CreateTeam	0.5 days	Fri 3/30/07	Fri 3/30/07
38	CreateRole	0.5 days	Fri 3/30/07	Fri 3/30/07
39	GrantRightCompanyAdministrator	0.5 days	Mon 4/2/07	Mon 4/2/07
40	LoginAdminSystem	0.5 days	Mon 4/2/07	Mon 4/2/07
41	LogoutAdminSystem	0.5 days	Tue 4/3/07	Tue 4/3/07
42	CreateTeam	0.5 days	Tue 4/3/07	Tue 4/3/07
43	LoginTrackingSystem	0.5 days	Wed 4/4/07	Wed 4/4/07
44	MaintainUser	0.5 days	Wed 4/4/07	Wed 4/4/07
45	ViewIssueList	0.5 days	Thu 4/5/07	Thu 4/5/07
46	ViewProfile	0.5 days	Thu 4/5/07	Thu 4/5/07
47	Database Design (Physical model)	3 days	Thu 3/29/07	Mon 4/2/07
48	Create Detail Design Document V0.1	3 days	Tue 4/3/07	Thu 4/5/07
49	**Implementation**	**5 days**	**Tue 4/3/07**	**Mon 4/9/07**
50	Create MySQL Database	2 days	Tue 4/3/07	Wed 4/4/07
51	MaintainUser	1 day	Fri 4/6/07	Fri 4/6/07
52	GrantRightCompanyAdministrator	1 day	Mon 4/9/07	Mon 4/9/07
53	CreateTeam	1 day	Fri 4/6/07	Fri 4/6/07
54	CreateDepartment	1 day	Mon 4/9/07	Mon 4/9/07
55	LoginAdminSystem	1 day	Fri 4/6/07	Fri 4/6/07
56	LogoutAdminSystem	1 day	Mon 4/9/07	Mon 4/9/07
57	**Test**	**3 days**	**Thu 4/5/07**	**Mon 4/9/07**
58	Test Database and detail design	1 day	Thu 4/5/07	Thu 4/5/07
59	Test first build	2 days	Fri 4/6/07	Mon 4/9/07



	Task Name	Duration	Start	Finish
60	**Construction phase**	**13 days**	**Tue 4/10/07**	**Thu 4/26/07**
61	**1st iteration**	**8 days**	**Tue 4/10/07**	**Thu 4/19/07**
62	**Analysis and design**	**4 days**	**Tue 4/10/07**	**Fri 4/13/07**
63	LogoutTrackingSystem	0.5 days	Tue 4/10/07	Tue 4/10/07
64	ColumnChartReportOfPhases	0.5 days	Tue 4/10/07	Tue 4/10/07
65	ColumnChartReportOfIssueStatus	0.5 days	Wed 4/11/07	Wed 4/11/07
66	ColumnChartReportOfArtifacts	0.5 days	Wed 4/11/07	Wed 4/11/07
67	ReportOfIssuePerProjectMember	0.5 days	Thu 4/12/07	Thu 4/12/07
68	ExportIssueToXML	0.5 days	Thu 4/12/07	Thu 4/12/07
69	Create Detail Design Document V0.1	4 days	Tue 4/10/07	Fri 4/13/07
70	**Implementation**	**8 days**	**Tue 4/10/07**	**Thu 4/19/07**
71	CreateProject	1 day	Tue 4/10/07	Tue 4/10/07
72	ShowCurrentSession	1 day	Wed 4/11/07	Wed 4/11/07
73	GrantRightProjectAdministrator	1 day	Thu 4/12/07	Thu 4/12/07
74	GrantRightApplicationAdministrator	1 day	Fri 4/13/07	Fri 4/13/07
75	CreateDomain	1 day	Mon 4/16/07	Mon 4/16/07
76	CreateSubSystem	1 day	Tue 4/17/07	Tue 4/17/07
77	AssignProjectRole	1 day	Wed 4/18/07	Wed 4/18/07
78	CreateRole	1 day	Thu 4/19/07	Thu 4/19/07
79	AssignProjectRight	1 day	Mon 4/16/07	Mon 4/16/07
80	ManagePhase	1 day	Tue 4/17/07	Tue 4/17/07
81	ManageIteration	1 day	Wed 4/18/07	Wed 4/18/07
82	ManageArtifact	1 day	Thu 4/19/07	Thu 4/19/07
83	LoginTrackingSystem	1 day	Mon 4/16/07	Mon 4/16/07
84	ManageIssue	1 day	Tue 4/17/07	Tue 4/17/07
85	ViewIssueList	1 day	Wed 4/18/07	Wed 4/18/07
86	ViewProfile	1 day	Thu 4/19/07	Thu 4/19/07
87	**Test**	**5 days**	**Fri 4/13/07**	**Thu 4/19/07**
88	Test build 2nd	3 days	Fri 4/13/07	Tue 4/17/07
89	Test build 3rd	2 days	Wed 4/18/07	Thu 4/19/07

	Task Name	Duration	Start	Finish	Predec
90	**2nd iteration**	**5 days**	**Fri 4/20/07**	**Thu 4/26/07**	
91	**Implementation**	**2 days**	**Fri 4/20/07**	**Mon 4/23/07**	
92	LogoutTrackingSystem	1 day	Fri 4/20/07	Fri 4/20/07	78
93	ColumnChartReportOfPhases	1 day	Mon 4/23/07	Mon 4/23/07	92
94	ColumnChartReportOfIssueStatus	1 day	Fri 4/20/07	Fri 4/20/07	82
95	ColumnChartReportOfArtifacts	1 day	Mon 4/23/07	Mon 4/23/07	94
96	ReportOfIssuePerProjectMember	1 day	Fri 4/20/07	Fri 4/20/07	86
97	ExportIssueToXML	1 day	Mon 4/23/07	Mon 4/23/07	96
98	**Test**	**5 days**	**Fri 4/20/07**	**Thu 4/26/07**	
99	Test build 4th	2 days	Fri 4/20/07	Mon 4/23/07	89
100	System test	3 days	Tue 4/24/07	Thu 4/26/07	99
101	**Transition phase**	**5 days**	**Fri 4/27/07**	**Thu 5/3/07**	
102	**1st iteration**	**5 days**	**Fri 4/27/07**	**Thu 5/3/07**	
103	**Analysis and design**	**5 days**	**Fri 4/27/07**	**Thu 5/3/07**	
104	Create user manual	5 days	Fri 4/27/07	Thu 5/3/07	100
105	Create installation guideline	2 days	Fri 4/27/07	Mon 4/30/07	100
106	**Implementation**	**3 days**	**Tue 5/1/07**	**Thu 5/3/07**	
107	Fix bug	3 days	Tue 5/1/07	Thu 5/3/07	105
108	**Test**	**3 days**	**Tue 5/1/07**	**Thu 5/3/07**	
109	Beta test	3 days	Tue 5/1/07	Thu 5/3/07	105

Figure 7.1- Project plan

8. Business Process model

8.1. BPM (Free form)

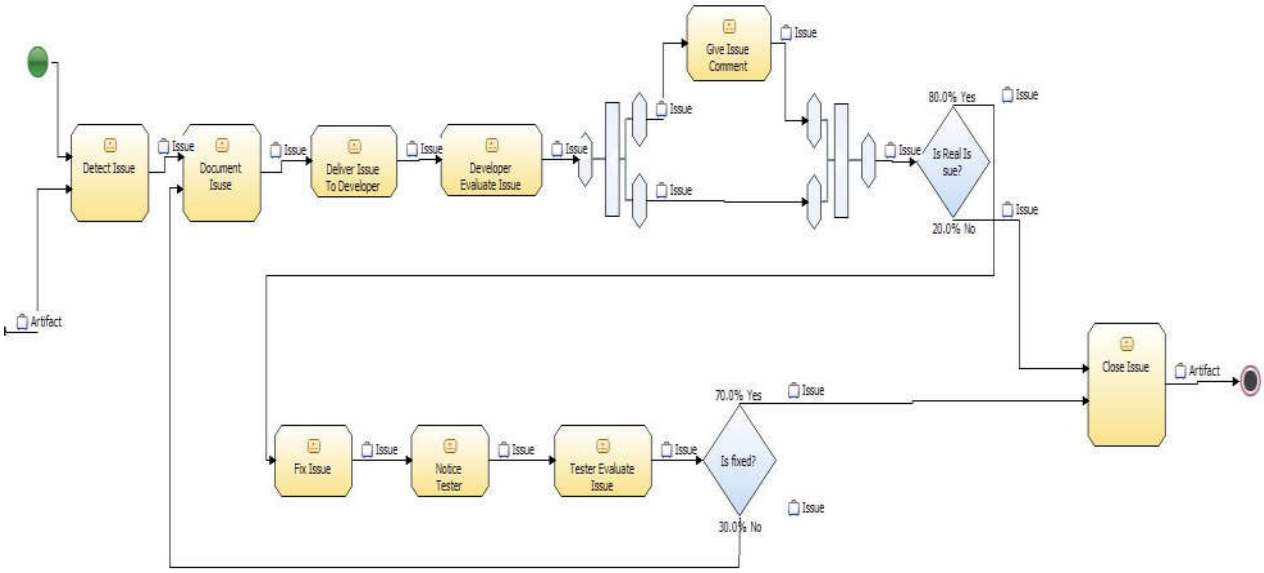


Figure 8.1- Business Process Model

8.2. Swim lane

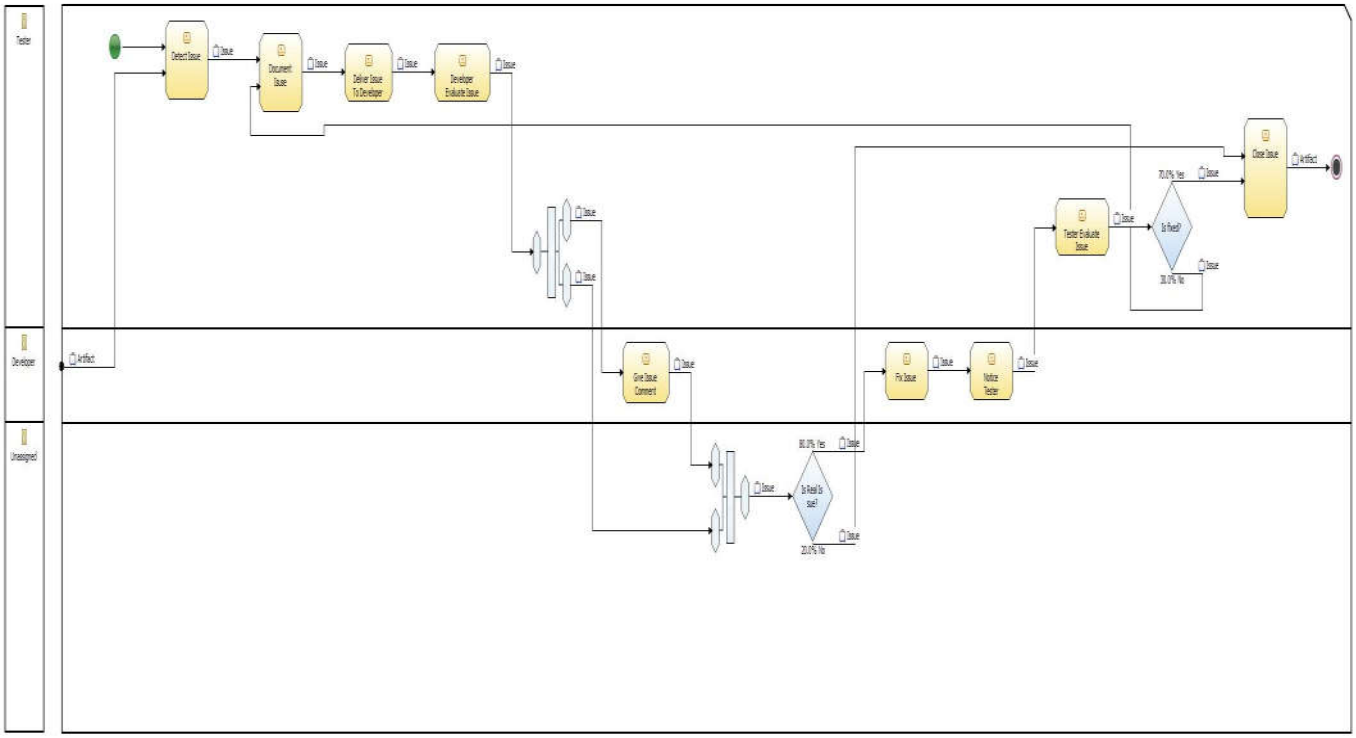


Figure 8.2- Swim lane model

8.3. Resource Catalog

Figure 8.3 - Resource catalog

8.4. Organization structure model

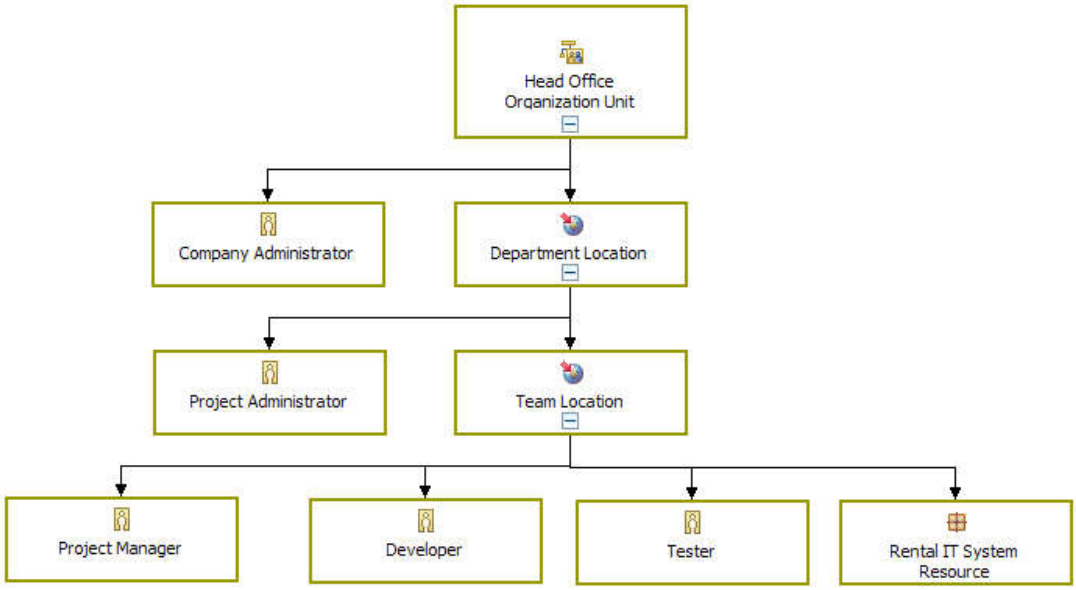


Figure 8.4 - Organization structure model

8.5. Model explanation

8.5.1. Task Description

Name	Role	Function	Input	Output
Detect Issue	Tester	Detect Issue and describe details about Issue	Artifact	Issue
Document Issue	Tester	List and report Issue	Issue	Issue
Deliver Issue to Developer	Tester	Assign this Issues to Developer	Issue	Issue
Developer evaluate Issue	Developer	Base on the Issue, Developer will evaluate each of them	Issue	Issue

Give Issue comment	Developer	Developer will add issue comment	Issue	Issue
Fork		Select one of two possible path, choose the first path if want to give comment.	Issue	Issue
Join		Input can come from two ways depending on if issue has comment	Issue	Issue
Is real Issue?		Decide path to move basing on correctness of issue. Usually 80% of issues detected correctly, 20% is false	Issue	Issue
Fix Issue	Developer	Developer receives Issue (if have) and fixes them	Issue	Issue
Notice Tester	Developer	When issue was fixed, Developer will notice to Tester	Issue	Issue
Tester evaluates Issue	Tester	Tester receives Issue that is fixed by Developer and revaluate it	Issue	Issue
Is fixed?		Decide path to move basing on correctness of fixing action. Usually 70% of cases, developer fix issue correctly, 30% is false	Issue	Issues
Close Issue	Tester	Close issue if it was fixed correctly or it was not a real issue	Issue	Artifact

8.5.2. Connection Description

"From" element	"To" element	Connection name	Associated data
External Boundary	Detect Issue	Process Start Connection	Artifact
Start	Detect Issue	Start Connection	None
Detect Issue	Document Issue	Document Issue Connection	Issue
Document Issue	Deliver Issue To Developer	Deliver Issue To Developer Connection	Issue
Deliver Issue To Developer	Fork	Fork Connection	Issue
Fork(1)	Give Issue Comment	Give Issue Comment Fork Connection	Issue
Give Issue Comment	Join(1)	Give Issue Comment Join	Issue

		Connection	
Fork(2)	Join(2)	Join Connection	Issue
Join	Is Real Issue	Is Real Issue Connection	Issue
Is Real Issue(Yes)	Fix Issue	Fix Issue Connection	Issue
Is Real Issue(No)	Close Issue	Close Issue Connection1	Issue
Fix Issue	Notice Tester	Notice Tester Connection	Issue
Notice Tester	Tester Evaluate Issue	Tester Evaluate Issue Connection	Issue
Tester Evaluate Issue	Is fixed	Is fixed Connection	Issue
Is fixed(Yes)	Close Issue	Close Issue Connection2	Issue
Is fixed(No)	Document Issue	Document Issue2	Issue
Close Issue	Stop Node	Stop Node Connection	Artifact

8.6. Business Process Model Analysis

Activity Name	Average Elapsed Duration	Average Delay Duration	Average Throughput
Close Issue	1 minute	0 seconds	60 work items / hour
Deliver Issue To Developer	3 minutes	0 seconds	20 work items / hour
Detect Issue	20 minutes	0 seconds	3 work items / hour
Developer Evaluate Issue	10 minutes	0 seconds	6 work items / hour
Document Isuse	15 minutes 46.666 seconds	0 seconds	3.803 work items / hour
Fix Issue	25 minutes 10 seconds	0 seconds	2.384 work items / hour
Fork	0 seconds	0 seconds	Invalid Value
Give Issue Comment	5 minutes	0 seconds	12 work items / hour
Is Real Issue?	0 seconds	0 seconds	Invalid Value
Is fixed?	0 seconds	0 seconds	Invalid Value
Join	0 seconds	0 seconds	Invalid Value
Notice Tester	5 minutes	0 seconds	12 work items / hour
Tester Evaluate Issue	10 minutes	0 seconds	6 work items / hour
ITS Process As-Is	1 hour 42 minutes 36 seconds	0 seconds	0.585 work items / hour

Figure 8.5- Activity duration analysis

Average time required to complete the process: 1 hour 42 minutes 36 seconds
That task that requires the most time to complete: 25 minutes 10 seconds

Activity Name	Average Revenue	Average Processing Cost	Average Idle Cost	Average Allocated Resource Cost	Average Total Cost	Average Profit
ITS Process As-Is	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00
Document Isuse	$0.00	$1.00	$0.00	$0.67	$1.67	($1.67)
Deliver Issue To Developer	$0.00	$0.20	$0.00	$0.10	$0.30	($0.30)
Developer Evaluate Issue	$0.00	$2.00	$0.00	$0.33	$2.33	($2.33)
Give Issue Comment	$0.00	$0.30	$0.00	$0.00	$0.30	($0.30)
Is Real Issue?	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00
Fix Issue	$0.00	$1.50	$0.00	$0.00	$1.50	($1.50)
Notice Tester	$0.00	$0.20	$0.00	$0.00	$0.20	($0.20)
Tester Evaluate Issue	$0.00	$0.50	$0.00	$0.33	$0.83	($0.83)
Is fixed?	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00
Close Issue	$10.00	$0.20	$0.00	$0.03	$0.23	$9.77
Detect Issue	$0.00	$1.00	$0.00	$0.67	$1.67	($1.67)
Fork	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00
Join	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00

Figure 8.6- Activity cost analysis

The average cost of each task: see Average Total Cost Column.
The most expensive task: Developer Evaluate Issue.

All cases are grouped into four categories:

Catagory1: The issue is really a bug and is fixed by developer in two passes. The issue is solved incorrectly in first pass to developer. Because developer fixed one bug twice, the cost for fixing bug is double. That explains for the negative value of average profit **($-3.23**)

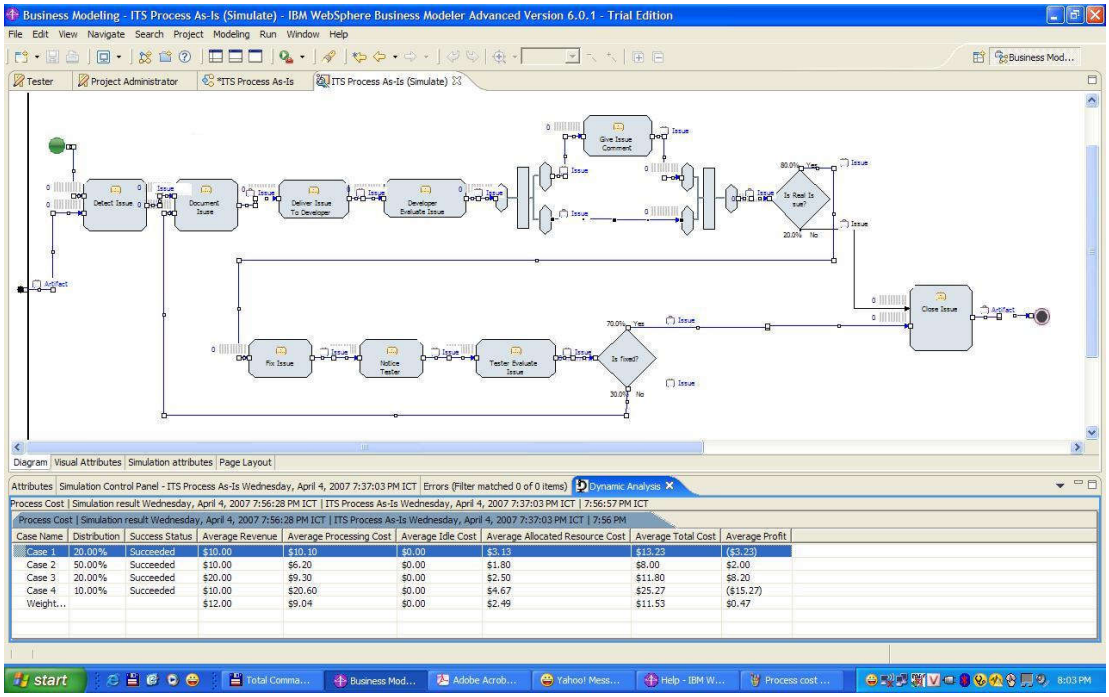


Figure 8.7- Process cost analysis – case 1

Catagory2: The issue is submitted by tester but developer verifies that it's not really a bug. There's no cost for fixing bug. The average profit is **$2**

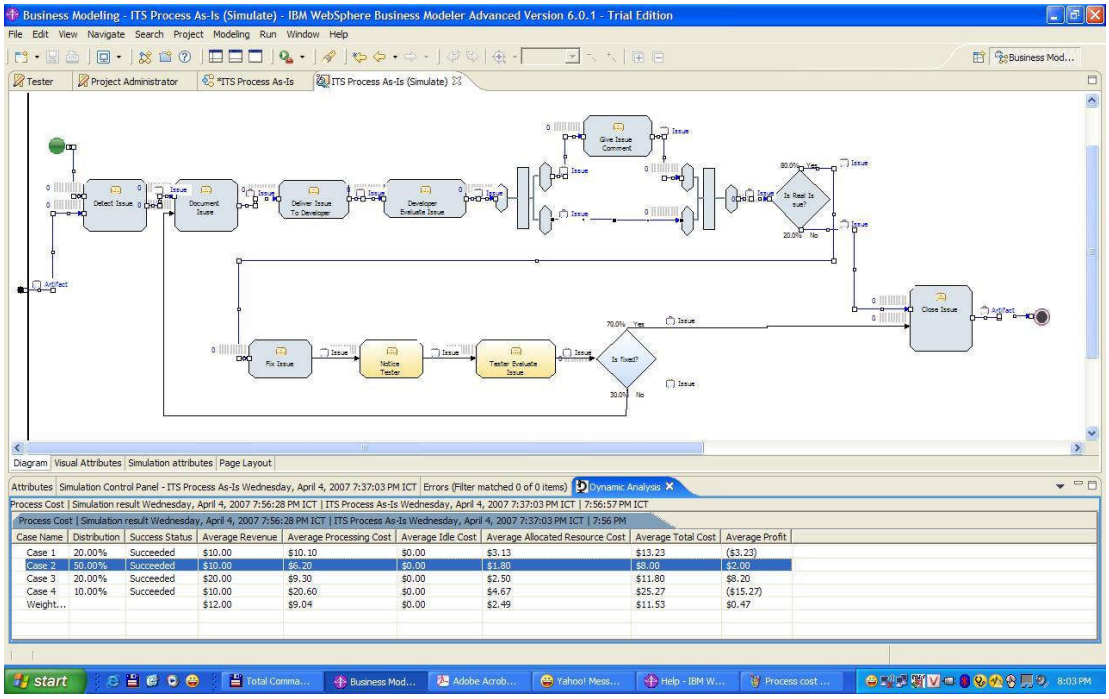


Figure 8.8- Process cost analysis – case 2

Catagory3: The issue is really a bug and is fixed by developer in one pass. Bug was fixed in the first time it was delivered to developer. The benefit of detecting and closing a real bug are high so the profit is positive **($8.2)**

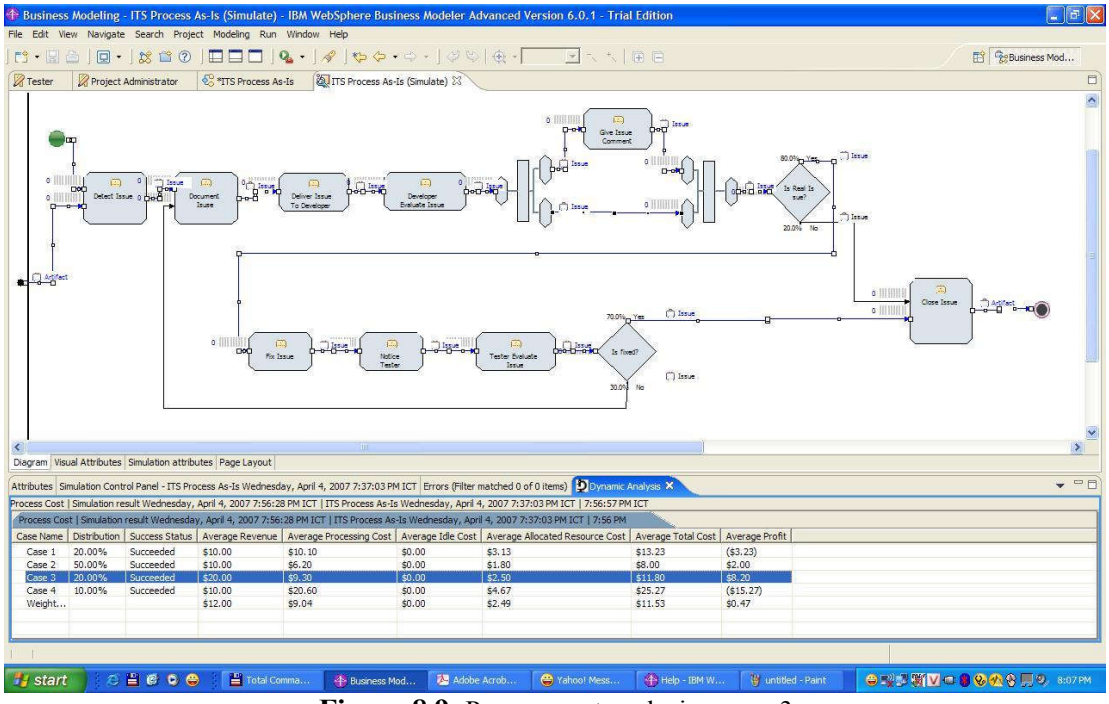


Figure 8.9- Process cost analysis – case 3

Catagory4: The issue was fixed by developer but when tester retest, he confirm that issue need to be fixed again and re-submit it to developer. Then developer verifies and sees that it was fixed and tester made mistake. The cost for re-testing and re-evaluate are high. Furthermore, the issue turns back and forward in many turns. That leads the profit is negative **($-15.27)**

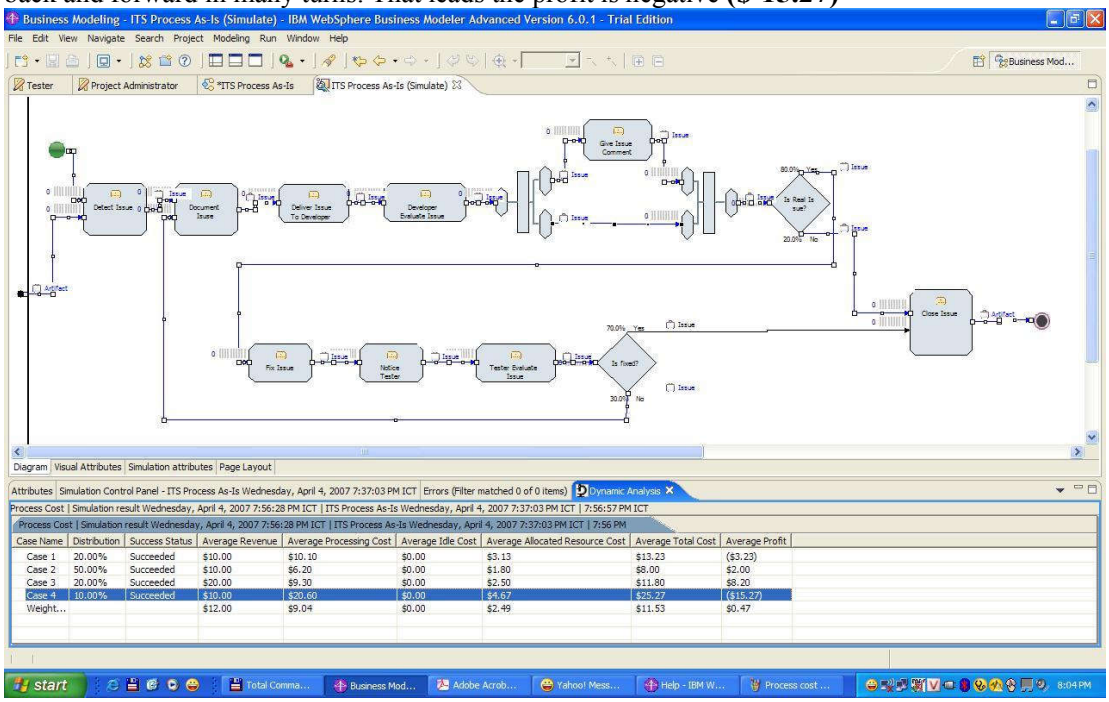


Figure 8.10- Process cost analysis – case 4

8.7. Data Flow Diagram

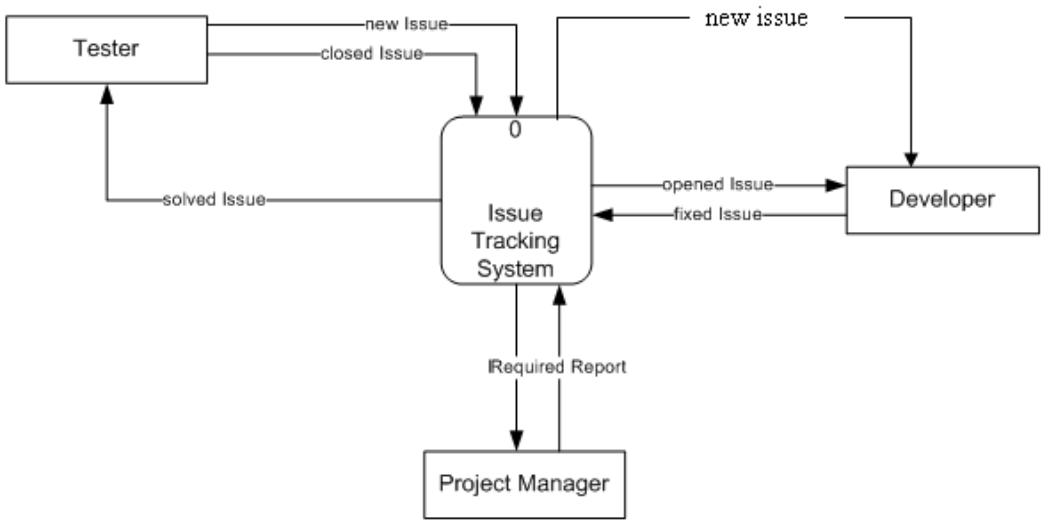


Figure 8.10- Context diagram of ITS

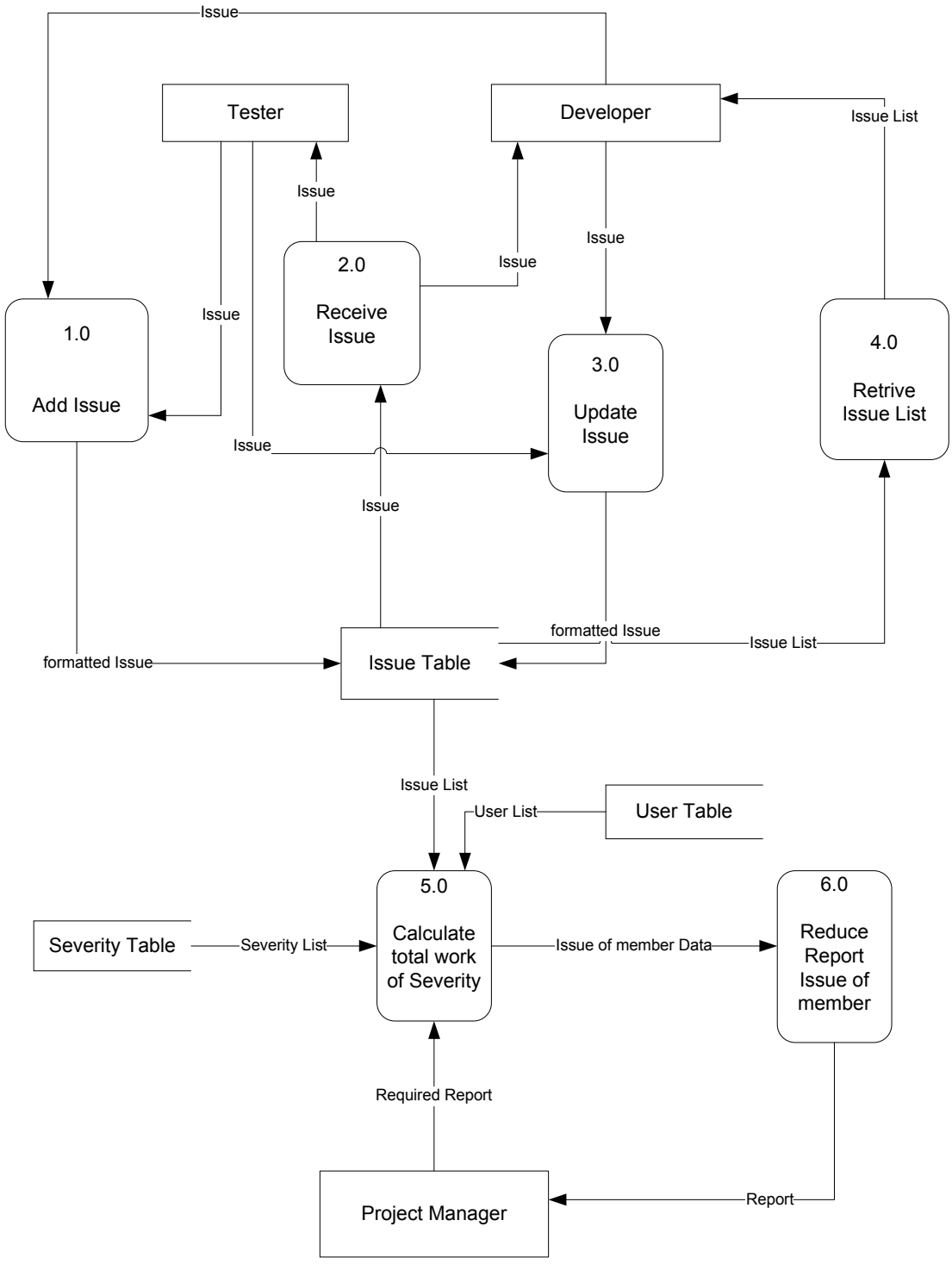


Figure 8.11- Level-0 DFD of ITS

9. Design

9.1. Database design

9.1.1. Conceptual Database Design

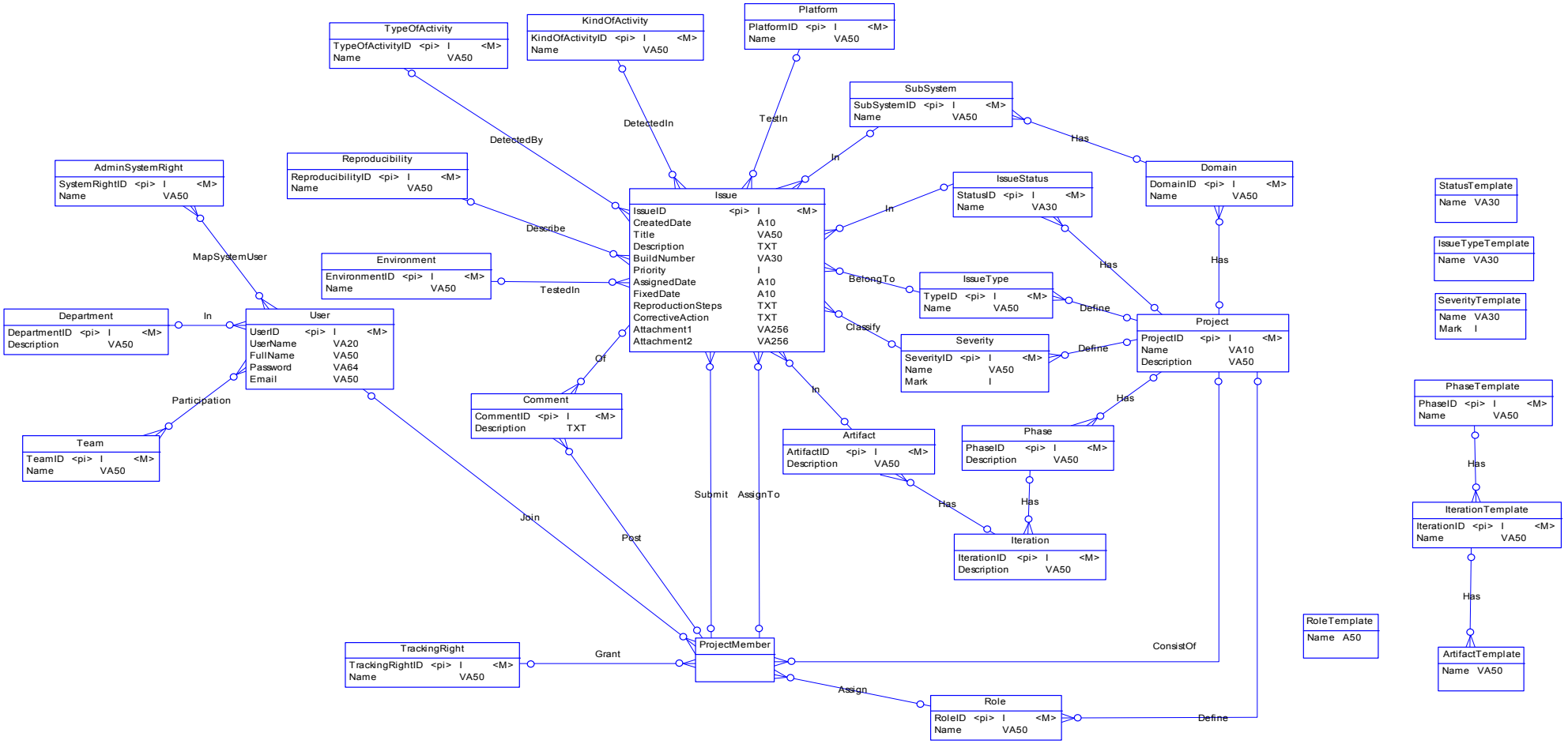


Figure 9.1- ITS Conceptual Data Model

9.1.2. Physical Database Design

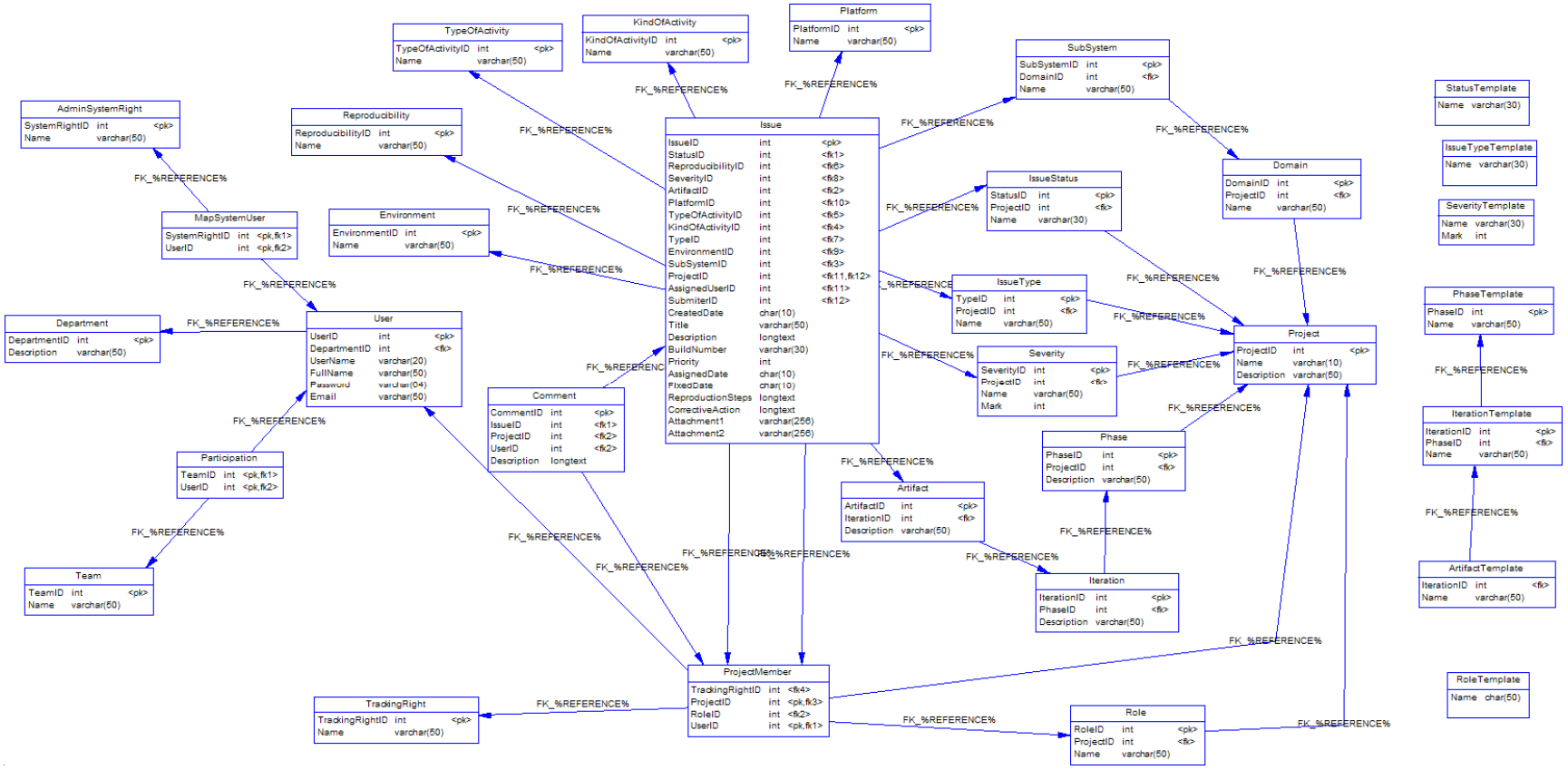


Figure 9.2- ITS Physical Data Model

9.2. Class Design
9.2.1. Class Diagram Overview

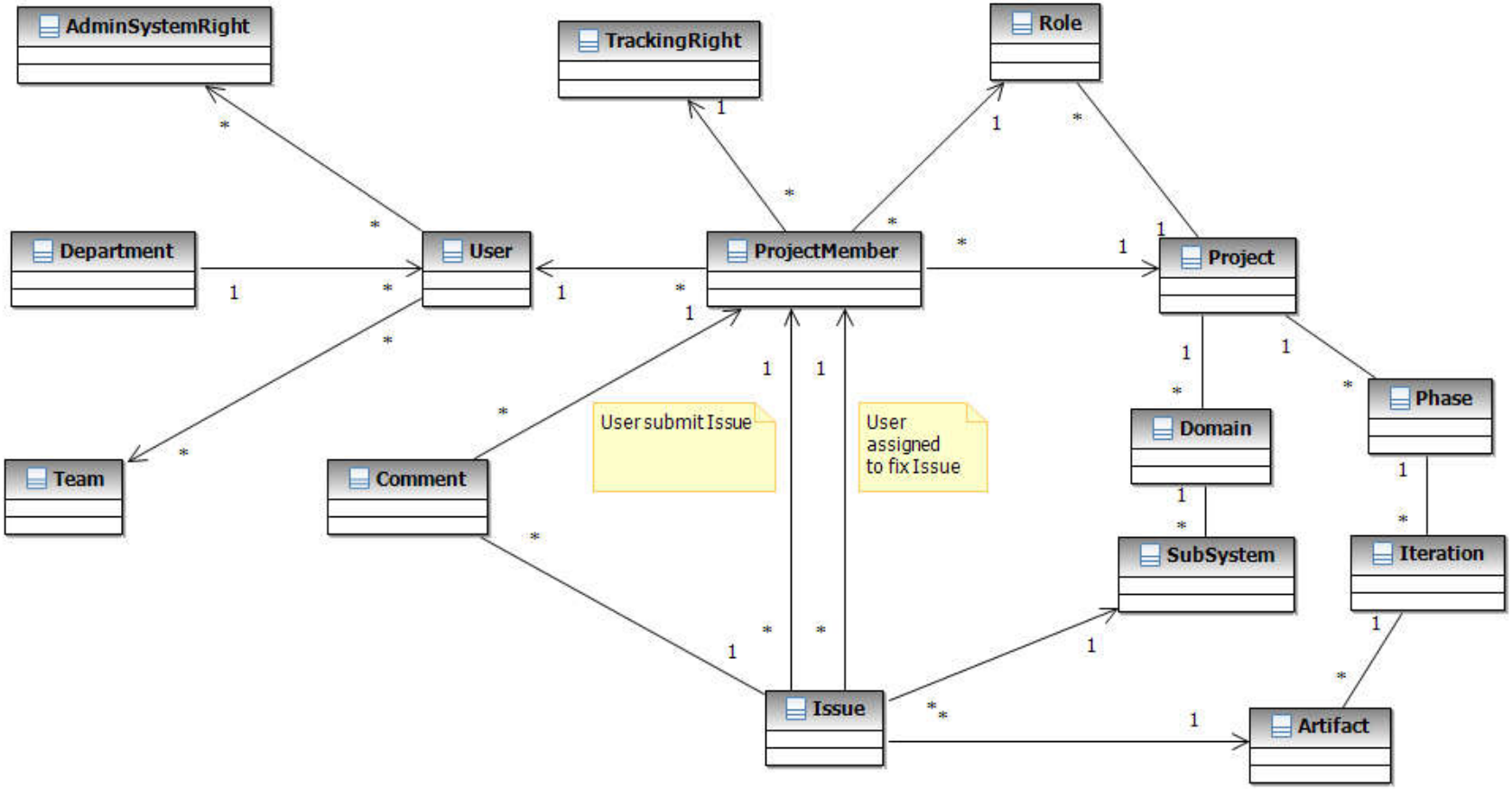


Figure 9.3- ITS Class Diagram Overview

9.2.2. Class Design Detail

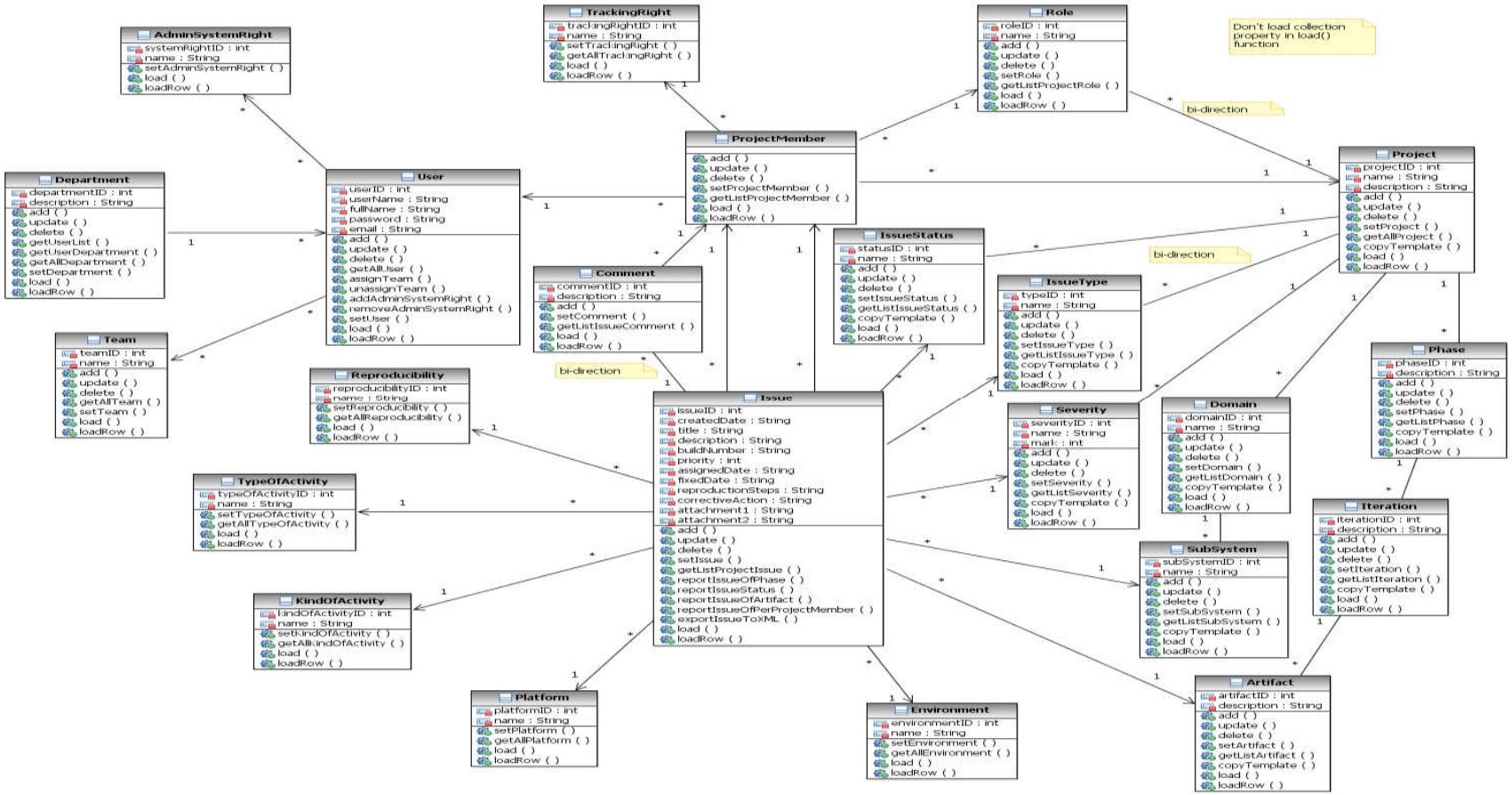


Figure 9.4- ITS Class Detail Design

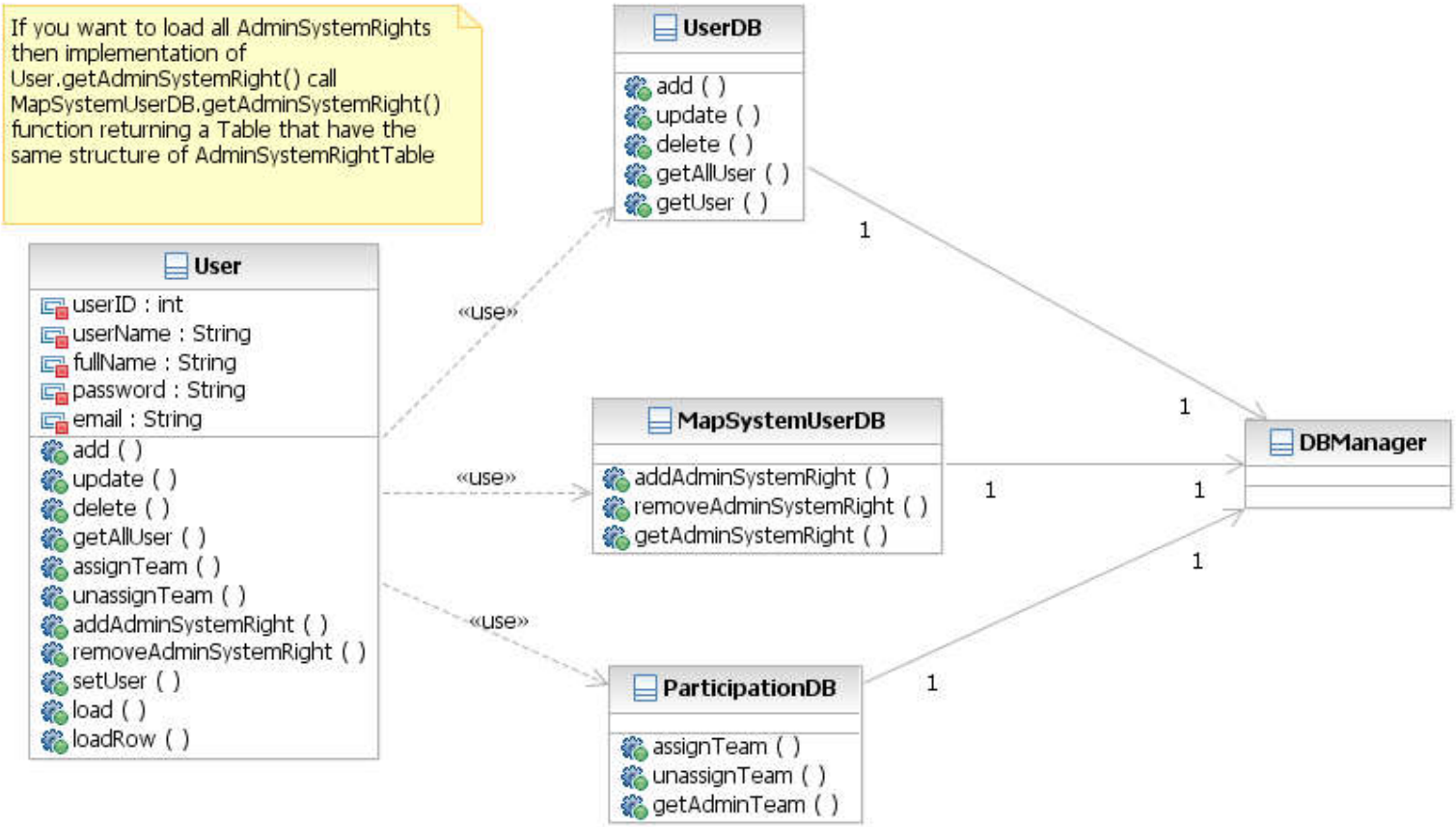


Figure 9.5- Map between User Class and Database

Figure 9.6- Map between Issue Class and Database

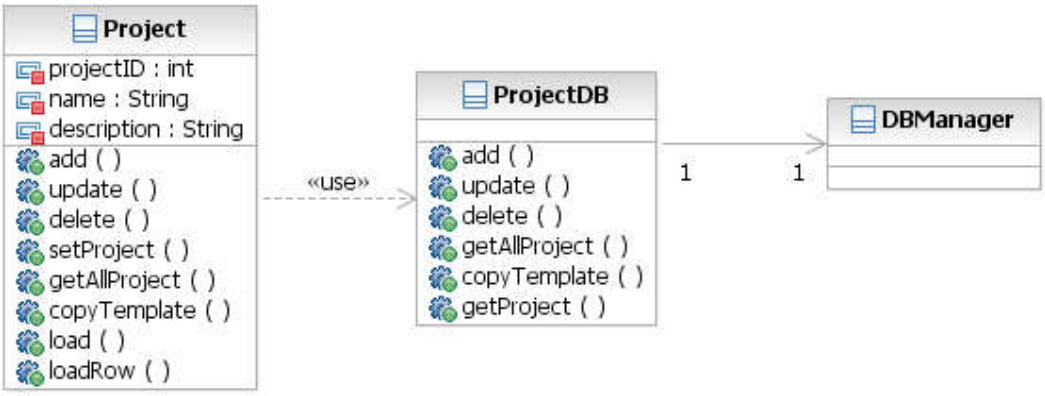


Figure 9.7- Map between Project Class and Database

9.3. Package Design

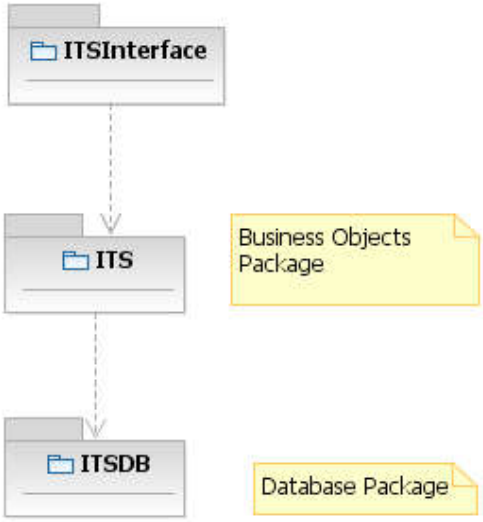


Figure 9.8- Package diagram

9.4. State Diagram

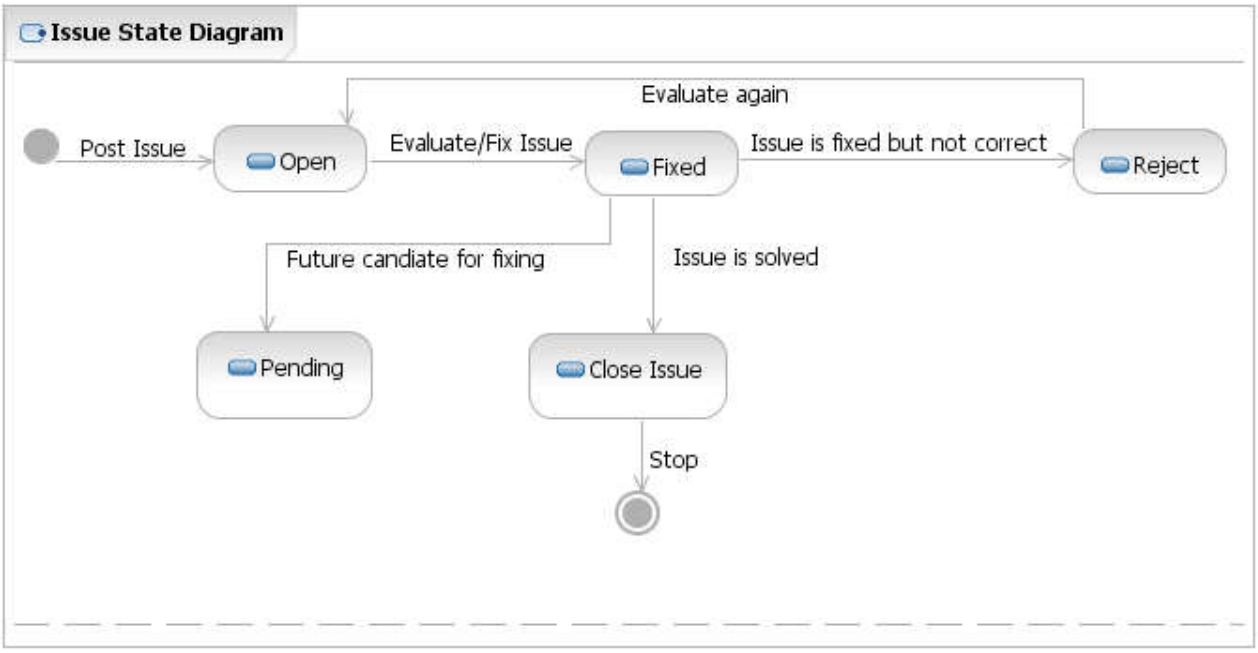


Figure 9.9- Issue State Diagram

9.5. Use case realization

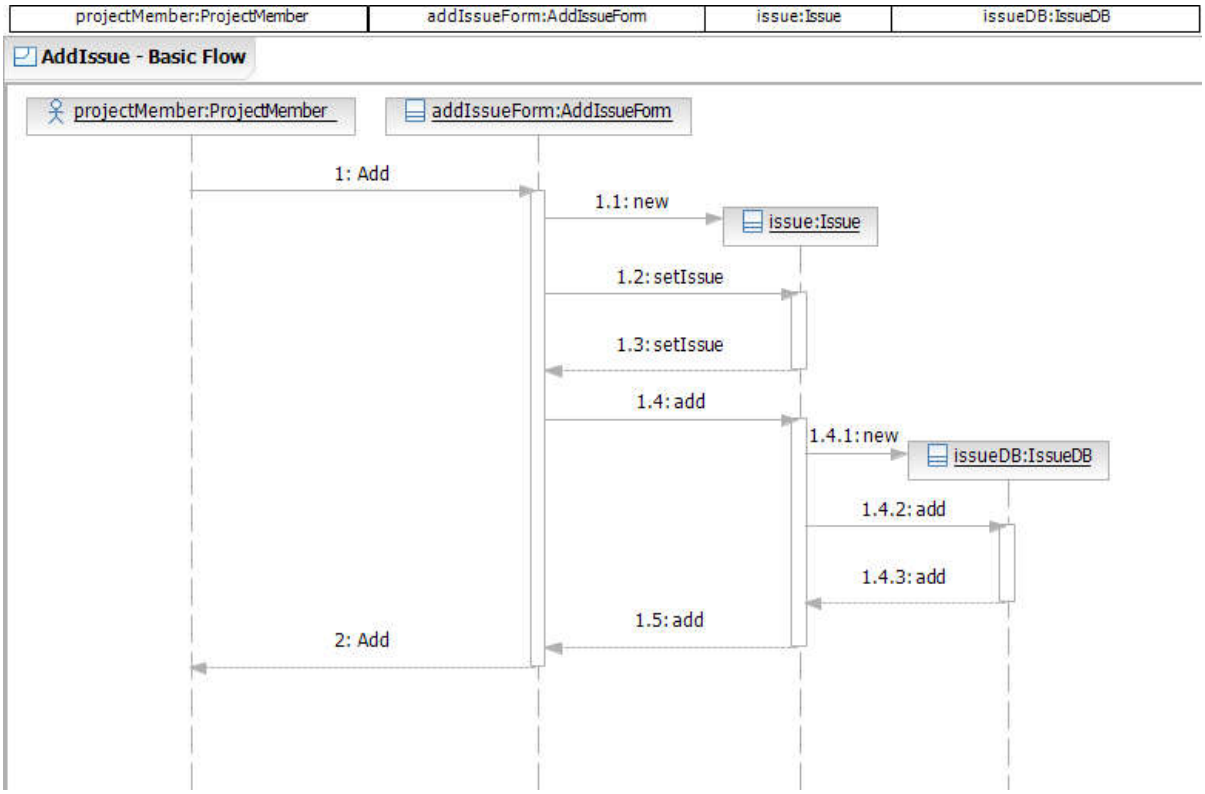


Figure 9.10- Add issue sequence diagram

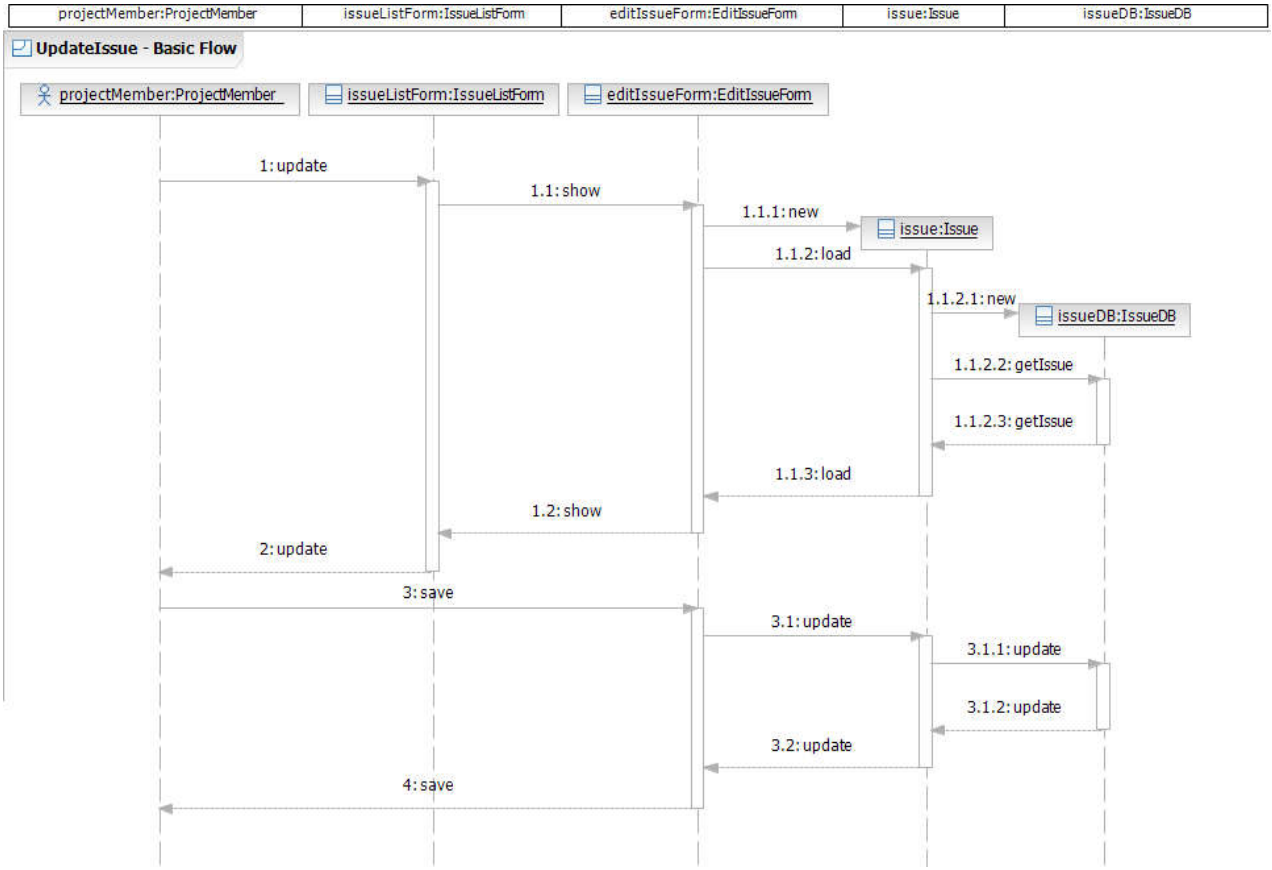


Figure 9.11- Update issue sequence diagram

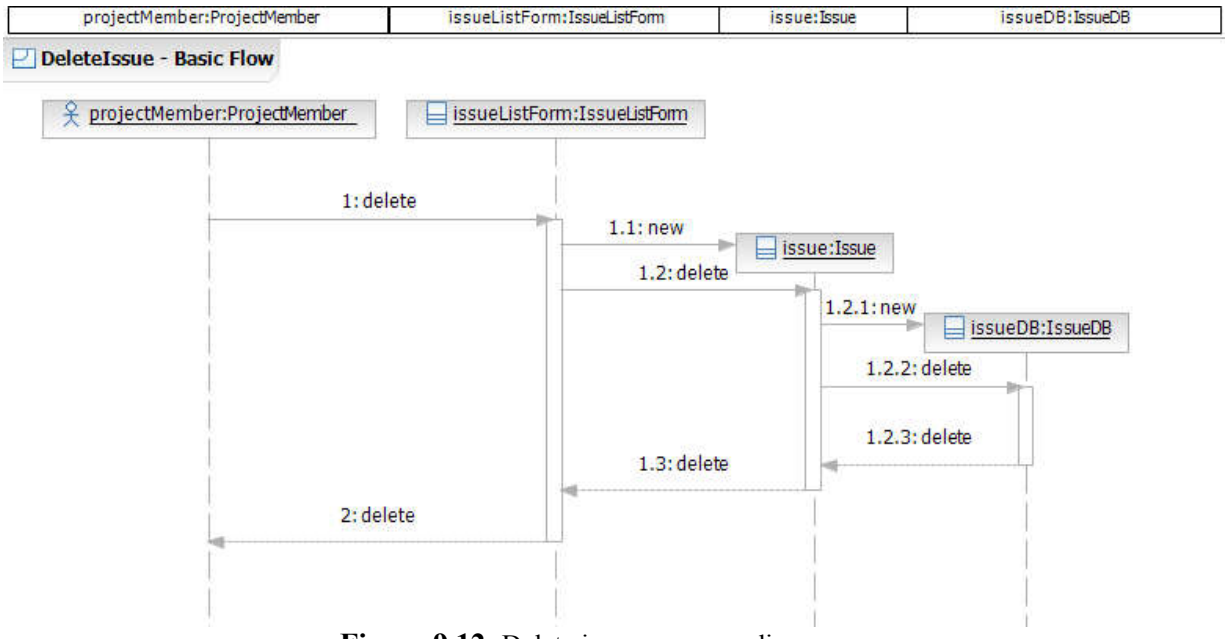


Figure 9.12- Delete issue sequence diagram

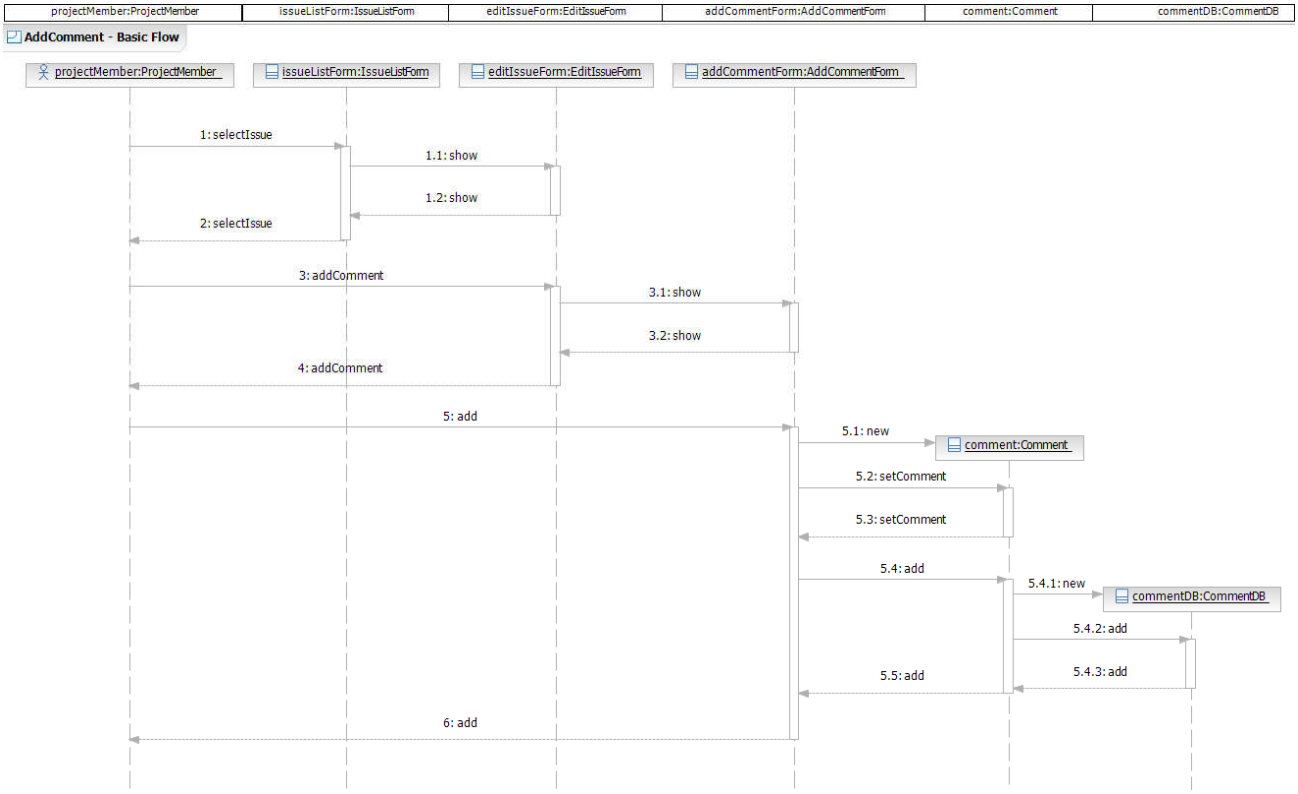


Figure 9.13- Add comment sequence diagram

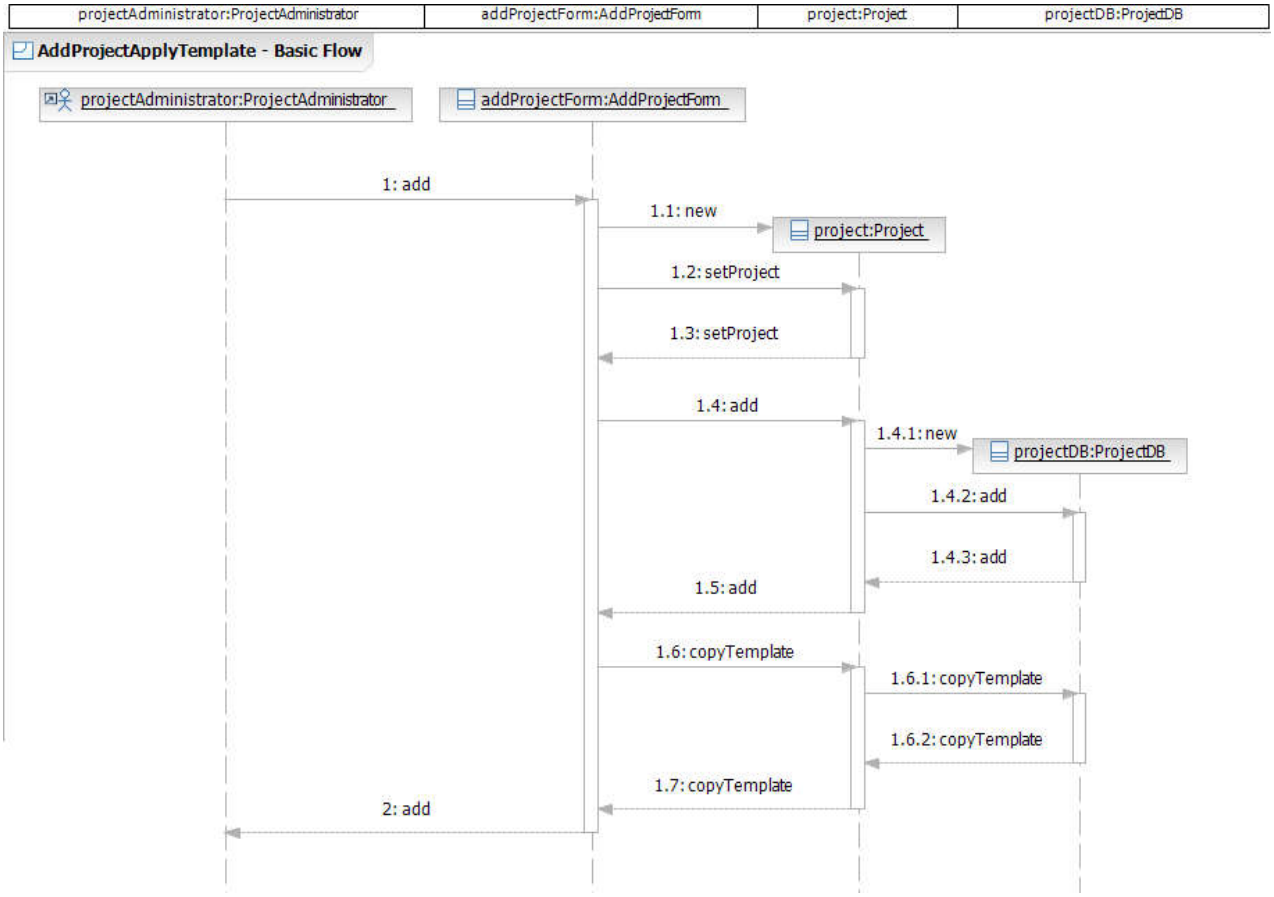


Figure 9.14- Add project apply template sequence diagram

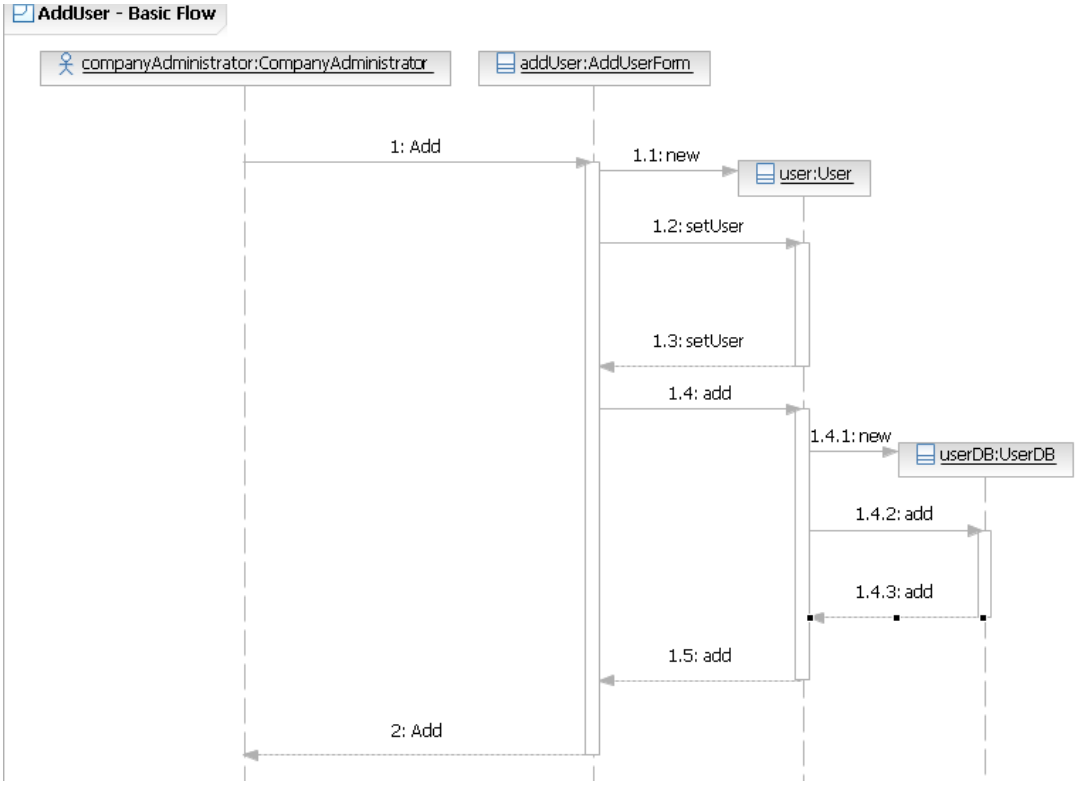


Figure 9.15- Add User sequence diagram

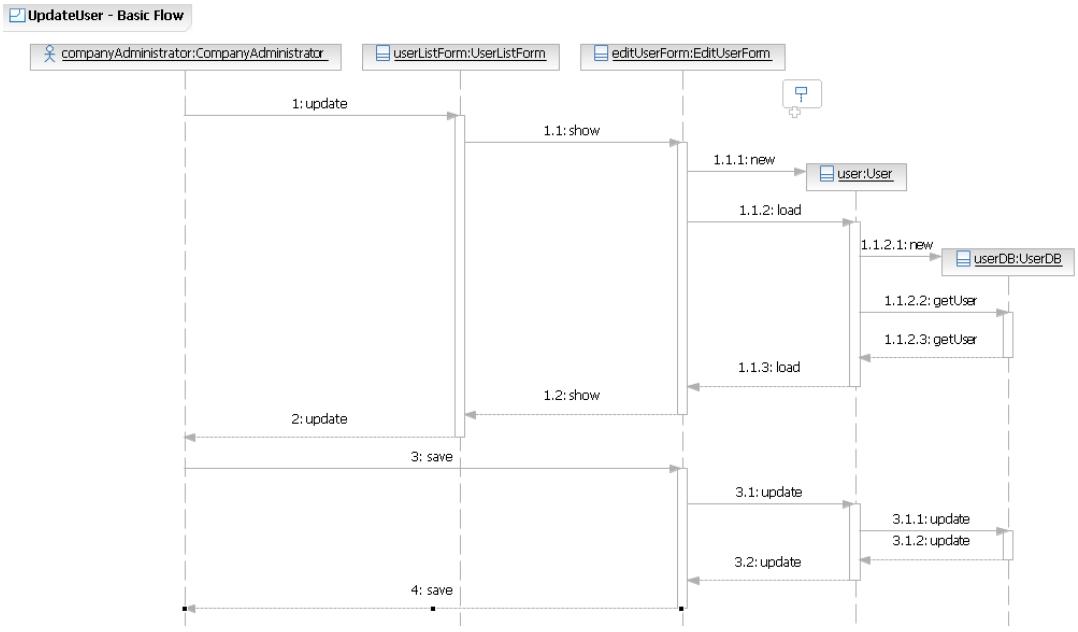


Figure 9.16- Edit User sequence diagram

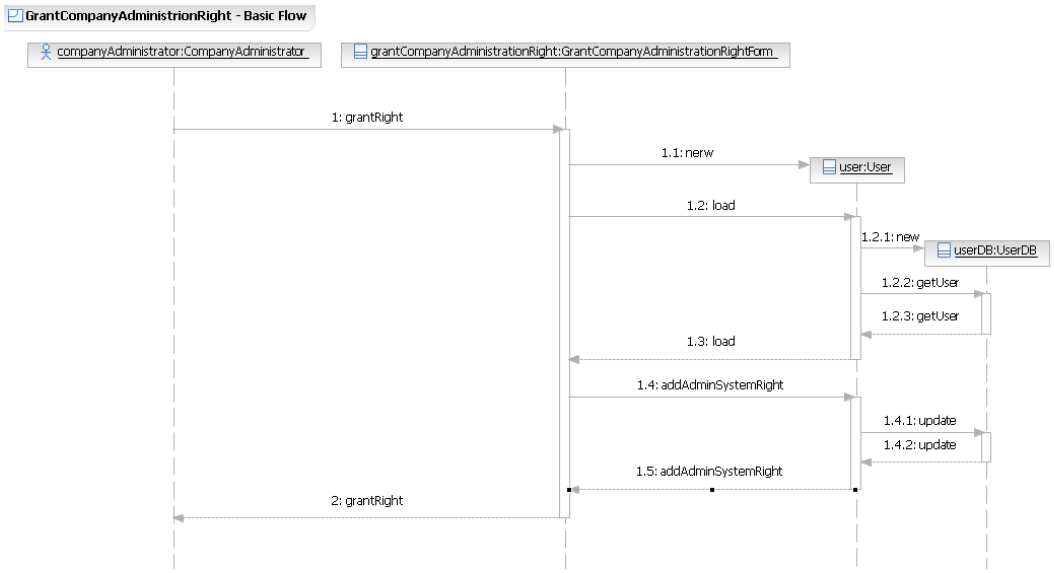


Figure 9.17- Grant Right of Company Administration sequence diagram

10. Documents for Testing

10.1. Test Plan

Please refer to a separate document **ITS - Test Plan.doc**

10.2. Test Case

Please refer to a separate document **ITS - TestCase.doc**

11. User Guidelines

Please refer to a separate document **UserGuideline.doc**

12. Installation Guidelines

Please refer to a separate document **Installation.doc**

13. Glossary

No.	Term	Definition
1	CompanyAdministrator	Someone who has highest right in system. This person can manage departments, teams, and users of system
2	ApplicationAdministrator	A person who follow application's activities, define project and project's administrator.
3	ProjectAdministrator	Someone who define project's information such as domain, subsystem, phase, iteration, artifact, and assign member to project.
4	NormalUser	Someone who take part in project. He can create project issue information.
5	ProjectManager	Someone who is responsible for managing his project.
6	Department	A department of company
7	Team	A group of user, that ProjectManager assign to projects
8	Role	A title that can be set to user
9	Right	A right, that can be set permission to users on their

		project
10	Project	A software project, that users can work issue tracking on
11	Issue	A document to describe information tracking of software project
12	Status	The status of Issue
13	Opened	The status of issue will be opened when user post new issue
14	Fixed	The status of issue be fixed when user change the status to fixed
15	Closed	The status of issue be closed when user change the status to closed
16	Pending	The status of issue be pending when user change the status to pending
17	Title	Title of issue
18	Description	Description for issue
19	Kind of Activity	Kind of activity testing on software project
20	Unit Test	A activity testing on each unit of software project
21	Integration Test	A activity testing on many units of software project
22	System Test	A activity testing on whole system
23	Type of Activity	Type of activity, that user did on software project
24	Test	A action test on software project
25	Review	A action review on software project
26	Phase	Phases of UP Architecture
27	Inception	A first phase on UP Architecture
28	Elaboration	A second phase on UP Architecture
29	Construction	A third phase on UP Architecture
30	Transition	A fourth phase on UP Architecture
31	Iteration	Iteration of phase such as: Analysis, design, construction, testing…
32	Artifact	An artifact on each phase of software project
33	Environment	A progress of software project
34	Development	A type progress of software project, when software project be in development
35	Pre-Production	A type progress of software project, when software project be pre-production
36	Production	A type progress of software project, when software project be production
37	Qualification	
38	Platform	A operating system, where software was installed for testing
39	Priority	A priority level, that was set to issue to describe the tracking on software project
40	Type of Issue	Type of Issue
42	Bug	A type of issue, which issue describe bug on software project
43	Change Request	A type of issue, which issue describe a changing request on software project
44	Evolution	A type of issue, which issue describe a evolution on software project
45	Suggested	A type of issue, which issue describe a suggestion on

		software project
46	Build Number	A ID number is assigned to issue
47	Fixed Date	The day, when issue be fixed
48	Reproduction Step	The steps, that were describe some steps to recognize problems of software project on issue
49	Corrective Action	The steps, that were describe some steps to fix problems of software project on issue

14. Project team

Our team consists of four members:
Vo Duc Thuan: project manager and interface designer.
Nguyen Ngoc Minh: developer and tester.
Ngo Cao Dinh: system analyst and developer.
Dao Anh Quan: business analyst and tester

15. Appendix

Meeting Minutes

Topic	Project Meeting		
Date	March 10, 2007	Time (from… to…)	
Place	Meeting Room No. 2		
Chair	Huynh Tan Sang	Recorder	Ngo Cao Dinh
Attendees	Huynh Tan Sang (Project manager), Pham Tien Phuc(Developer), Le Thi My Hien (Tester), Vo Duc Thuan (Interviewer)		
Non-Attendees			

Items Discussed:

- Tester's activities relating to an issue.
- Developer's activities relating to an issue
- Project manager's activities relating to an issue and project quality.
- Current process
- Expectation.

Details:

a. Tester

- Describe the detected issue clearly and assign the corrective responsibility to a person.
- The content of issue which he must record include following fields:
  - Title of issue
  - Description: describe details about issue
  - Kind of activity: describe QC activity such as Unit Test, Integration Test, system Test.
  - Type of activity: describe how this issue is detected. For example: Test software, review document, code, design, etc.
  - Phase: describe which phase this issue detected.

- Iteration: describe which iteration this issue detected
- Artifact: describe which artifact contain this issue
- Domain: describe which domain this issue belongs to
- Subsystem: describe which subsystem this issue belongs to
- Assign to: assign this issue to a suitable person
- Assign date: describe the date when tester detect this issue
- Reproducibility: describe how often this issue appears. For example: all days, impossible to reproduce, N/A, random, sometimes, undetermined.
- Reproduction steps: describe steps to reproduce this issue.
- Environment: describe the environment where this issue occurs. For example: Development, pre-production, production, qualification.
- Platform: describe which platform this issue was tested on. For example: UNIX, Solaris, and Window.
- Priority: describe the priority of this issue: Low, Medium, High
- Severity: describe the severity of this issue: Minor, Major, Critical, Fatal
- Type: describe kind of issue such as bug, suggested, change request, evolution.
- Build number: describe which build contains this issue.
- Attach file: attach image file displaying this issue.
- Status: describe status of this issue: Open, Fixed, Reject, Closed, pending.

b. Developer

- Can submit an issue like Tester
- Can view all fields of issue but can only change two fields:
    - Corrective Action: describe the actions performed to solve this issue.
    - Fixed date: the date this issue is fixed.

c. Project Manager

- Have all rights of Tester and Developer
- View reports belonging to the project manager right:
    - Report issues of each phase. This report is presented as column chart.
    - Report issues according to status of issue. This report is presented as column chart.
    - Report issues according to a selected artifact. Show severity of issues. This report is presented as column chart.
    - Report issues of a specific project member.

d. Current Process

There is no available system supporting for following the issues of project. When a tester detects a new issue, he has to send an email or talk directly to a developer to explain about the issue. The developer who fixed the issue announce to tester so that he can re-test and confirm that this issue was solved. When a project manager want to have a report about the issues to analysis, testers have to collect information from old email or just write down what he still remember. This process leads to a lot of problems and damage to the quality of the software product due to following reasons:

- The issues have not been stored fully and consistently. Therefore, an issue may get loss and it will not be solve early. It is also hard to collect all old issues for report and analysis.
- Lacking a standard way to describe an issue leads misunderstanding between the submitter and the solver.
- Lacking capability to show quickly the quality status of project.

c. Expectation

Company had a plan to improve software development process. The new process will adapt from Unify Process. Therefore, the test process also has to be improved to make sure that the quality of project can be controlled more effectively. Users in soft development department expect the new system may have following new features:

- Manage issues of each phase and artifact. There're many kinds of issues. Some of them are bugs, others are revolution, bug, suggested, change request.
- Help facilitate management of users and roles in a project. Each user has a specific role and right in project.
- Help each project member address his work quickly. Testers can see how many issues need to re-test and close. Developers can see which issues assigned to them.
- Help project managers follow the quality status of project by giving suitable reports described above.
- Have ability to export issue list to XML files so that information exchange among stakeholders is easier.