CHAPTER 4: GRAT Analysis and Design

4.1 Introduction

This chapter looks into the analysis and design of GRAT. The former identifies the entire functional and non-functional requirements. An analysis using object-oriented analysis is presented. The design part also uses object-oriented design for the architecture of the system.

4.2 GRAT Analysis

4.2.1 Requirements Analysis

This section is broken into two sub divisions, functional and non-functional requirements. Functional requirements are functionality of GRAT system, and non-functional requirements describe other aspects such as usability, efficiency and other runtime properties.

4.2.1.1 Functional Requirements

There are nine modules that were identified for the project. Six are based on Ian Sommerville’s (1996) requirements analysis method as mentioned in Chapter Two, one is to support his model, another as a project repository or database and the last one is to support scheduling. The modules are,

- Project Repository
- Domain Understanding
- Requirements Collection
CHAPTER 4: GRAT Analysis and Design

- Categories Collection
- Classification
- Conflict Resolution
- Prioritization
- Validation
- Activity Scheduling

There system also supports three roles. One is a facilitator as mentioned at the last part of Chapter Three who will be called, as Project Manager, Domino Administrator and the others will be identified and Team Members.

GRAT is a web-based requirements analysis tool. Meaning, by default it is a client-server web application. Clients are required to have a web browser and the server is the host computer of the organization.

**Project Repository**

i. The system lists all the projects the organization is involved.

ii. The system lists the project title, the relevant PROJECT MANAGER, START DATE, END DATE and PROJECT WEB SITE.

iii. The system provides links for all the projects.

iv. The system opens the PROJECT DETAILS page of the double-clicked project on the list. The PROJECT DETAILS page displays the name of the project, start date, end date, team members, details of the project and a link to the project web site. It has a button to return to the PROJECT REPOSITORY page.

v. The system provides a button and function for NEW PROJECT and DELETE PROJECT.
vi. The system only allows ADMINISTRATOR to use DELETE PROJECT.

vii. The NEW PROJECT button opens up a form that requires the PROJECT MANAGER to fill in the information. The user who clicks the New Project form is the PROJECT MANAGER. The information needs to be filled are name of the project, start date, end date, team members, details of the project. The start date is auto-generated from the server and it should display the current date. It shall have three buttons that are TEAM MEMBERS to select the team members, REGISTER to register the project and ABORT to ignore the information and open up the Project REPOSITORY page. The System checks to see all the information has been filled in. The System submits the form to the ADMINISTRATOR when the PROJECT MANAGER clicks the REGISTER button.

viii. The ADMINISTRATOR approves the project and provides an automated URL for the project. The ADMINISTRATOR can relocate the URL to other than the generated URL. Approved project has the website listed in the PROJECT REPOSITORY page.

**Domain Understanding**

i. TEAM MEMBERS are able to share information with other TEAM MEMBERS and PROJECT MANAGER via web form.

ii. TEAM MEMBERS are able to submit the forms on text basis, attach a file, or create a hyperlink.

iii. The system allows all TEAM MEMBERS to look at the information in the page.

iv. TEAM MEMBERS are able to choose to read the information by double clicking on the selected list.
CHAPTER 4: GRAT Analysis and Design

v. The system provides a button for TEAM MEMBERS to submit the information they want to share.

Requirements Collection

i. TEAM MEMBER are able post the requirements that will be feasible for the domain.

ii. All TEAM MEMBER are able to see the whole set of requirements submitted to the server.

iii. The system refreshes the list of submitted requirements every 5 seconds or whenever a new requirement is posted.

Categories Collection

i. TEAM MEMBERS are able to post the appropriate categories while looking and browsing into the list of requirements.

ii. The system refreshes list of categories every 15 seconds or when the TEAM MEMBER posts a new category.

Classification

i. TEAM MEMBER matches each requirements with appropriate category.

ii. The list of categories is provided in the list box.

iii. The system checks if all the requirements have been matched with an appropriate Category.

iv. The System previews the classification result of each TEAM MEMBER based on the categories if all the requirements have a matching category. TEAM MEMBERS who is not satisfied with their choice are allowed to modify their selection.
v. The system allows TEAM MEMBERS to submit their classification result.

vi. The system calculates and computes the overall result from the submission of the TEAM MEMBER. The calculation and computation is based on voting process.

**Conflict Resolution**

i. The system allows TEAM MEMBER to see the requirements sorted out according to the categories.

ii. The system allows users voice opinion for a particular requirement.

iii. The system displays a form for TEAM MEMBER to voice out conflict regarding a requirement by double clicking.

iv. The system allows responds to that conflict by clicking a button. The conflicts and the responds to the conflicts are arranged in a hierarchy form.

**Prioritization**

i. TEAM MEMBERS are able to prioritize each requirement in their respective categories.

ii. The values of prioritize are numbered. The range depends on the number of requirements in each category.

iii. The system provides the values in list box. 1 represent highly prioritize and 6 will represent the least prioritize. The value is converted to score.

iv. The system checks if all the requirements in a category have a unique value.

v. The system previews the prioritization result of each TEAM MEMBER based on the categories if all the requirements have a unique priority. TEAM MEMBERS who is not satisfied with their choice are allowed to modify their selection.
vi. The system computes and totals the overall score from the submission of the TEAM MEMBER. The calculation and computation is based on voting process. The lowest scored requirement will be recorded as highest priority.

**Validation**

i. The system allows TEAM MEMBERS who feel that a particular requirement has some problem, to double click on the requirements, and prompt TEAM MEMBER to the requirements validation form.

ii. The system displays the submitted result displayed below the respective requirements.

**Activity Scheduling**

i. The system displays the scheduled activities for the current week and following week in the project web site.

ii. The system provides a page to submit new activity. TEAM MEMBER has to fill in the date of the activity is suppose to be held and activity description. The default date is the current date of the system and it can be edited. The system provides auto text generation for the requirements analysis activities, which are Domain Understanding, Requirements Collection, Categories Collection, Classification, Conflict Resolution, Prioritization and Validation.

**Project Manager**

i. PROJECT MANAGER is able to change the project status and set the current project status. The statuses are Domain Understanding, Requirements Collection, Categories Collection, Classification, Conflict Resolution, Prioritization and Validation.
ii. PROJECT MANAGER is able to delete the materials collected at all activities.

**Domino Administrator**

i. DOMINO ADMINISTRATOR is able to provide URLs for the respective project either by using default values or edit the values.

### 4.2.1.2 Non-Functional Requirements

The non-functional requirements are considered when building GRAT in addition to the above mentioned functional requirements. The requirements are stated in the following sub sections.

**Usability**

The system must be built taking into considering the groupware activities. It should take into consideration different level of users and thus provide the respective tools that support the respective users.

**Maintainability and expandability**

The system must be easily updated, maintained and expanded from time to time without affecting the current data and/or information.

**Portability**

In order to make the system as a web application, clients only require a web browser without worrying of the platform. The system must be aimed to be running on Microsoft Internet Explorer version 5.5 and above.
CHAPTER 4: GRAT Analysis and Design

Efficiency

GRAT must be able to provide a good response time. Being a web-based application, GRAT should consider minimizing response, process and page reloading time.

Scalability

GRAT should support scalability, meaning that the services of the system must be provided regardless of the size of the system. There should not be a limitation or whatsoever on the number of requirements or categories, which mean the system, should support a small scale or a big scale project.

4.2.2 Object-Oriented Analysis

For this research, the GRAT was developed using object-oriented analysis and design with traditional implementation. The main benefit using this approach is being able to model the problem domain using OO analysis. OO analysis promises a more natural medium for communication between end user and the analysts than current structured analysis techniques provide (Wilkie, 1993).

The Unified Modeling Language or UML (UML, 2000) was used to represent the system's diagrams. The analysis and design steps presented in Appendix A are based on the reference made to S.R.Schach (Schach, 1999) The diagrams are presented in Appendix A.
4.3 GRAT Design

4.3.1 GRAT Architecture

GRAT is a client server application. The system is a two-tier system. Meaning that only two layers will be interacting with each other. The two layers are the presentation layer which runs in the client and the application layer which runs in the server.

4.3.1.1 The Presentation Layer

Since the system is running on the web, the client is triggered by the web browser. The browser will get connected to the application that runs on a web server. It will be the interface for interactions between users and the application running background.

4.3.1.2 The Application Layer

The GRAT application resides on the server. It binds to a certain IP address waiting for clients requests and respond to it. It should be able to handle all synchronous and asynchronous jobs.

4.3.2 Object-Oriented Design

Object-oriented design complements object-oriented analysis (Wilkie, 1993). Therefore, the transition from analysis to design is straightforward. A full object-oriented design will consist of a generalized and optimized class hierarchy design along with an application design based on the class design. Since GRAT will be applying
CHAPTER 4: GRAT Analysis and Design

traditional method for the implementation stage, this research has only included the class design and collaboration diagrams. The diagrams can be seen in Appendix A.

4.4 Summary

As seen and observed, this chapter presents the analysis and design specifications of GRAT. It was stared by stating the functional and non-functional requirements, followed by analysis, which was done using object-oriented methodology. Use case diagrams were developed at this stage. The second part looked into the design aspect of GRAT. It also contains the general architecture of GRAT. Again, using object-oriented modeling techniques, several class diagrams and interaction diagrams is presented. The diagrams are presented in Appendix A for the sake of simplicity.