Software Design Specification

For Online Library System

16.04.2012

Developed By Dogan Gunes and Yakup Arslan

Contents

1. Introduction

1.1 Purpose

This document will define the design of the Online Library System. It contains specific information about the expected input, output, classes, and functions. The interaction between the classes to meet the desired requirements is outlined in detailed figures at the end of the document.

1.2 Scope

This Design Specification is to be used by Software Engineering and Software Quality Engineering as a definition of the design to be used to implement the Online Library System.

1.3 Objective

The Online Library should help people to find what their needs about anything. Users of Online Library will create their own network by follow other users who share exactly what their interests in the different category which user creates to help to find out related topics under these categories. By categorizing, Users will reach the source of info faster with efficiency of time.

1.4 References

Software Design Description Template-Samet Tonyali
www.eksisozluk.com
http://www.vordweb.co.uk/specification.htm
http://www.uml-diagrams.org/examples/online-shopping-example.html

2.0 System Overview

2.1 Product Perspective

The website is a way to take information informally or formally. It also includes the idea or opinion of people that are interested about any topic. Recently, there exists many online website that give the information about everything but still there are lots of missing things in their system. They don't look at user's perspective. We are trying to look at the user's viewpoint and find out what actually they want and according to this observation, we are making our website.

2.1.1 Design Perspective

The design of this product utilizes an Object-Oriented Approach.

2.1.2 User Interface

The user of this system will interact with Topic, Search, Group and Comment part. In each part, Users will face with some options that related with their properties such as like, dislike, share … The System will allow user to check the topics, write comments, the users that are followed etc…

2.1.3 Hardware Interfaces

The System will work all computers that connects internet. Besides, when we develop an application for mobile phones, it will be active on phones as well.

2.1.4 Software Interfaces

This simulator will execute on Windows environment running on Visual Studio Compiler.

2.2 Product Functions

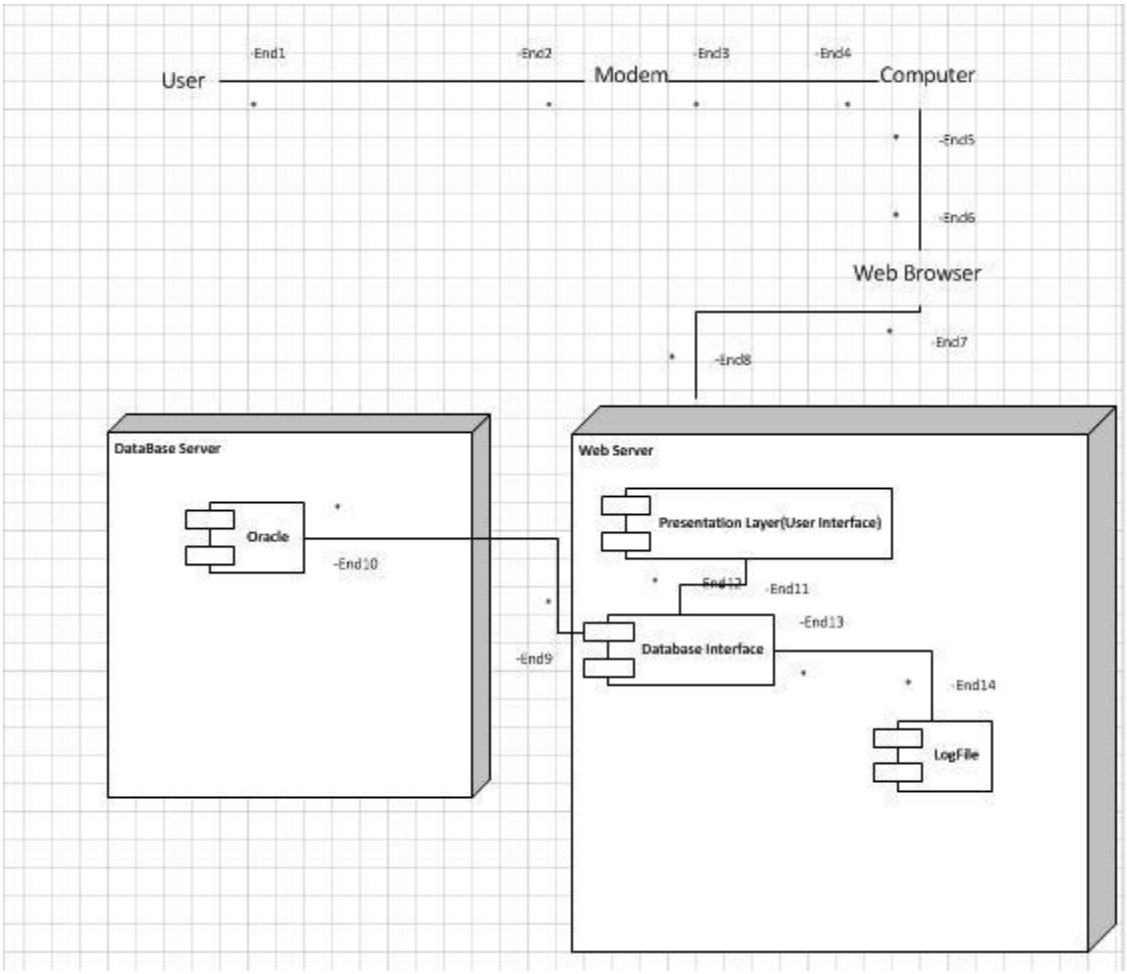
2.3 User Characteristics

Guests- that are allowed to use website, but they have some limitations such as not to make any comments etc.

Users- are able to use the website without any limitation; each of them has an account on it.

Admin- will able to control the website and monitor the comments and titles.

3.0 Deployment Diagram



4.0 Architectural Design

4.1 Introduction

In the AD phase, the software requirements are transformed into definitions of software components and their interfaces, to establish the framework of the software. This is done by examining the SRD and building a 'physical model' using recognized software engineering methods. The physical model is used to produce a structured set of component specifications that are consistent, coherent and complete. Each specification defines the functions, inputs and outputs of the component the major software components are documented in the Architectural Design Document (ADD). The ADD gives the developer's solution to the problem stated in the SRD. The ADD must cover all the requirements stated in the SRD (AD20), but avoid the detailed consideration of software requirements that do not affect the structure.

The main outputs of the AD phase are the:

• Architectural Design Document (ADD);

• Software Project Management Plan for the DD phase;
• Software Configuration Management Plan for the DD phase;
• Software Verification and Validation Plan for the DD phase;
• Integration Test Plan.

Progress reports, configuration status accounts and audit reports are also outputs of the phase. These should always be archived. To minimize risk of ıncompleteness and error, AD should include user representatives, system engineers, and hardware engineers and operations personal. Progress against plans should be continuously monitored by project management and documented regularly.

4.2 Construction of Physical Model

A software model is:

• A simplified description of a system;
• Hierarchical;
• composed of symbols organized according to some convention;
• built with the aid of recognized methods and tools;
• used for reasoning about the software.

A 'simplified description' is obtained by abstracting the essentials and ignoring non-essentials. A hierarchical presentation makes the model simpler to understand. A recognized method for software design must be adopted and used consistently in the AD phase. The key criterion for the 'recognition' of a method is full documentation. The availability of tools and training courses are also important.

The physical model:

• defines the software components;
• is presented as a hierarchy of control;
• uses implementation terminology (e.g. computer, file, record);
• Shows control and data flow.

The physical model provides a framework for software development that allows independent work on the low-level components in the DD phase. The framework defines interfaces that programmers need to know to build their part of the software. Programmers can work in parallel, knowing that their part of the system will fit in with the others.
The logical model is the starting point for the construction of the physical model. Design constraints, such as the need to reuse components, may cause departure from the structure of the logical model. Top-down design starts by defining the top level software components and then proceeds to define the lower-level components. The top-down approach is vital for controlling complexity because it enforces 'information hiding' by demanding that lower level components behave as 'black boxes'. Even if a design method allows other approaches (e.g. bottom-up), the presentation of the design in the ADD must always be top-down.
The construction of the physical model should be an iterative process where some or all of the tasks are repeated until a clear, consistent definition has been formulated.
In all but the smallest projects, CASE tools should be used for building a physical model. They make consistent models that are easier to construct and modify.

4.3 Decomposition of the software into components

The design process starts by decomposing the software into components. The decomposition should be done top-down, based on the functional decomposition in the logical model. The correspondence between functional requirements in the SRD and components in the ADD is not necessarily one-to-one and frequently isn't. When there are two or more feasible solutions to a design problem, the designer should choose the most cost-effective one. Correctness at each level can only be confirmed after demonstrating feasibility of the next level down. Such demonstrations may require prototyping. Designers rely on their knowledge of the technology, and experience of similar systems, to achieve a good design in just a few iterations.

The decomposition of the software into components should proceed until the architectural design is detailed enough to allow:

- Individual groups or team members to commence the DD phase;
- The schedule and cost of the DD and TR phases to be estimated.

In a multitasking real-time system, the appropriate level to stop architectural design is when the processing has become purely sequential.

4.4 Implementation of non-functional requirements

The specification of each component should be compared with all the requirements. This task is made easier if the requirements are grouped with their related functions in the SRD. Some kinds of non-functional requirements do not influence the architecture and their implementation can be deferred to the DD phase.

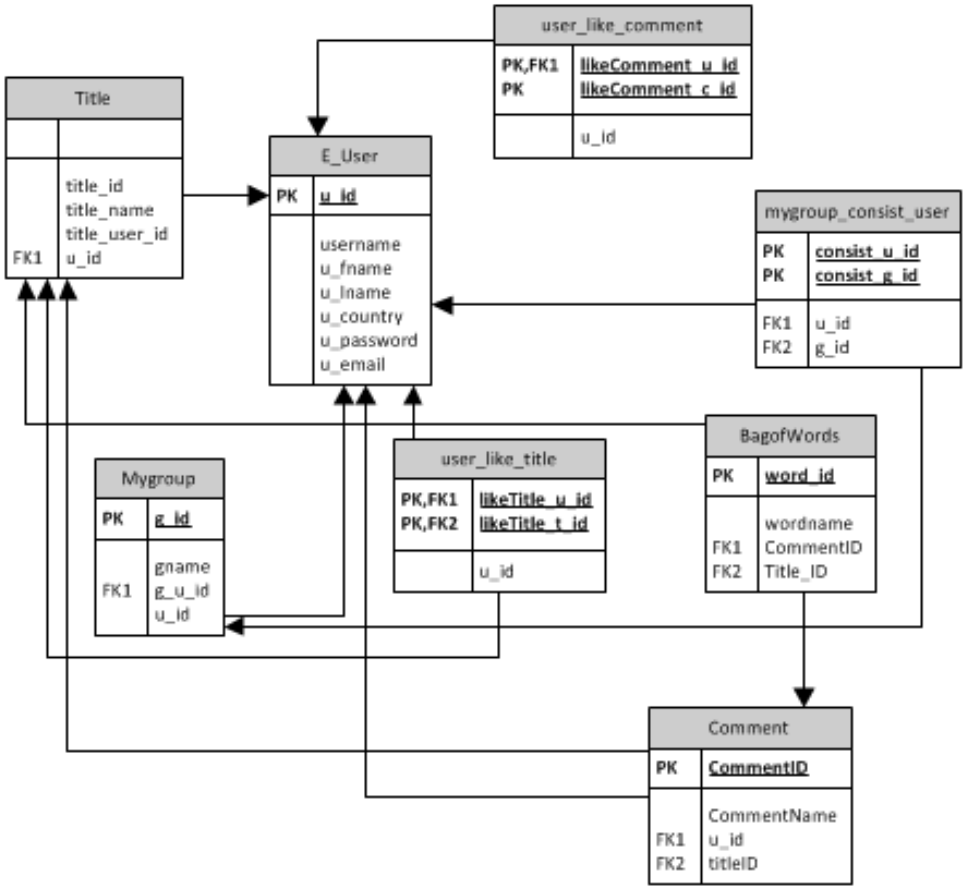
4.4.1 Performance Requirement

The system must response in admire time interval, fetching data from server or searching data should not make wait user too long. Since the Library will run on internet it is depend on internet speed but beside speed system should be fast.

3.4.2 Scalability Requirement

Server should be adoptive against increasing number of user, for example for 1000 user response time is 1 sec, for next user it should increase response time for acceptable.

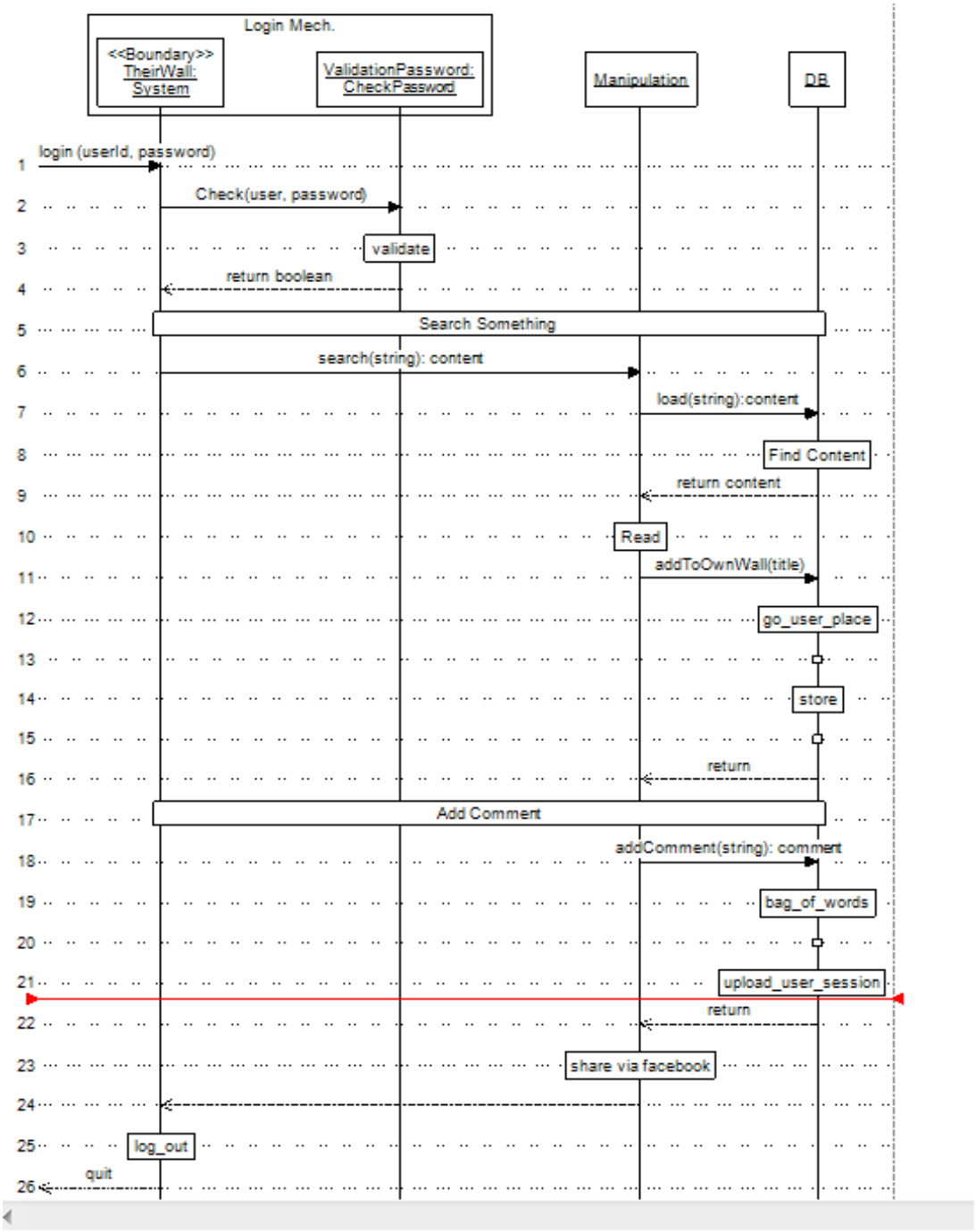
5.0 Data Structure Design



6.0 Use Case Realization

6.1 Sequence Diagram of Make Comment

I will show user login's behavior on system at figure below. After user browse the link address of system, user should type user name with corresponding password, so system can validate user's password before user login. If password is validated, user continues session as a member. At line 5, user search for something, system load specific data from database to user workspace. User can read this data or just copy to paste somewhere else or if user found this information really useful, user could save this content in his area (so it will be visible on his wall). At line 17, after user read some material under title, user can add comment but before comment is added, bag_of_procedure run to check whether content contain any non-proper word use, or not. If content is proper, then database upload changes. And user can share comment via Facebook or via other social media.



7.0 Real-Time Design

In our website (system), we have a control unit for the monitor comments and topics which has slang expressions. So BagofWords controlling system will check the words in the BagofWords table and if the system found the word in the database, it will delete the comment or the title immediately. That means, real-timing is important for the system, because we will not allow the bad words included in the website.

8.0 Help System Design

There will be help section in the design, it will provide users to use website efficiently. It will give some questions that users ask frequently and the answers will be written on the page. It will be like the other web site FAQ. We also include some video tips, and screenshot to visualize helping information.