**ITERATION #3**

# Software Design & Architectures

## SOFE 3650 Case Study: Course Management System (CMS)

**CRN: 43963**

**Prepared By:**

| | |
|---|---|
| 100617713 | Samantha Husack |
| 100620049 | Alexander Hurst |

**Due:** December 5, 2018
**Instructor:** Professor Ramiro Liscano
**Possible Marks for this section:** 15

# Case Study: Course Management System (CMS)
# ITERATION 3

## Table of contents

ADD Step 1: Review Inputs

1: Step 2: Iteration Goal By Selecting Drivers
2: Step 3: Choose Elements of the System to Refine
3: Step 4: Choose Design Concepts that Satisfy Selected Drivers
4: Step 5: Instantiate Architectural Elements, Allocate Responsibilities, Define Interfaces
5: Step 6: Sketch Views and Record Design Decisions
6: Step 7: Perform Analysis of Current Design and Review Iteration Goal

## Design Process

The following section reflects iteration 3 of the ADD process. First, we review the inputs. This is essential to identify the requirements chosen to be drivers in design decisions through the design process. Next, we go through this final iteration of the ADD Process for steps 2 through 7.

### ADD Step 1: Review Inputs

In the first step of the ADD method, we review the inputs and identify which requirements will be considered as drivers. These inputs are summarised in the following table.

| **Category** | **Details** |
|---|---|
| Design purpose | The purpose is to produce a sufficiently detailed design to support the construction of the system. |
| Primary functional requirements | From the uses cases presented in Assignment 2, the primary ones were determined to be:<br>UC-2: Because it directly linked to the core features of the system.<br>UC-5: Because of the technical issues associated with it.<br>UC-6: Because it directly linked to the core features of the system.<br>UC-9: Because of the technical aspects, as well as core functionality. |
| Quality attribute scenarios | The prioritized quality attributes below were chosen as drivers. |

| **ID** | **Importance to CMS Stakeholders** | **Implementation Difficulty** |
|---|---|---|
| QA-1 | HIGH | MEDIUM |
| QA-3 | HIGH | HIGH |
| QA-5 | HIGH | HIGH |
| QA-7 | HIGH | HIGH |
| QA-8 | HIGH | HIGH |

| **Category** | **Details** |
|---|---|
| Constraints | All of the constraints discussed in Assignment 2 are included as drivers. |
| Architectural concerns | All of the architectural concerns discussed in the previous iterations continue as drivers in this section. |

## Iteration 3

The third iteration of the ADD process aims to start fulfilling some quality attributes after building the fundamentals in iterations 1 and 2.

### 1: Step 2: Iteration Goal By Selecting Drivers

For this iteration, we will focus on refining the system to fulfil some of the quality attributes of the system. In this iteration, we will focus on the following drivers:

- QA-1: Security
- QA-3: Performance
- QA-7: Simplicity
- QA-8: Efficiency, Usability, Simplicity

### 2: Step 3: Choose Elements of the System to Refine

The elements that will be refined are the physical elements defined in iteration 1:

- Web Client
- Web Server
- Logic Tier Layer

### 3: Step 4: Choose Design Concepts that Satisfy Selected Drivers

The design concepts used in this iteration are the following:

| **Design Decision and Location** | **Rationale and Assumptions** |
|---|---|
| Introduce a *Limit Access Protocol* to the Cross-Cutting Module | Control what and who may access which parts of the system |
| Introduce a *Self-Test Procedure* to Detect Faults | This procedure enables a component to test itself for the correct operations, ensuring the performance requirements are met in terms of operational scale. |
| Introduce a *Locate Discover Service* to locate services via the known services | This service allows the system to locate services by searching through known directory services. This will enable the secondary systems to be found and utilised. This will also enable future interoperability of the system. |
| Add a *Tailor Interface* to the Web Client | This feature will allow for the adding and removing of capabilities to an interface such as translation, buffering and data streaming. This will ensure that requests are correctly handled for requests to secondary services. |
| Introduce a *Revoke Access Protocol* to the Logic Tier | This protocol allows us to limit access to sensitive resources, even for normally legitimate users and uses if an attack is suspected by the system. This will allow for further account security for all system stakeholders. |
| Introduce an *active redundancy tactic* by replicating the | By replicating the critical elements, the system can withstand the failure of on the replicated elements without affecting |

| | |
|---|---|
| application server and other critical components | functionality. This will also help to ensure less frequent and shorter downtimes for data needs and can be used to increase performance as load balancers. |
| Introduce an *intuitive interface* to the Web Client | By introducing an intuitive interface we allow users to experience a consistent UI with an intuitive descriptive UI with the possibility of descriptive text that can be read by screen reader software for the visually impaired. This will ensure a user-friendly interface. |

**4: Step 5: Instantiate Architectural Elements, Allocate Responsibilities, Define Interfaces**
The instantiation design decisions are summarized in the following table:

| **Design Decision and Location** | **Rationale and Assumptions** |
|---|---|
| Implement the ability to export to ICS and CSV for calendars | The system will be able to export the schedules and due dates in a format that can be imported by most calendars systems. |
| Use active redundancy and load balancing in the application server | Because two replicas of the application server are active at any time, it makes sense to distribute and balance the load among the replicas. This can be achieved by introducing the Load-Balanced Cluster Pattern.<br>This introduces the architectural concern 5: Manage State in Replicas. |
| Implement the Access protocol into the webserver and the Logic tier layer | The web server will avoid sending information about things that the user cannot access with their permissions and the logic tier will prevent the user from gaining access to the information and data that they do not have access to |
| Implement a Locate Method in the Logic tier layer | The Locate method will allow the system to become discoverable to other systems as well as find other locatable systems within the known directories. |
| Implement self testing modules into the web server + logic layer | The web server will check Itself and the logic layer for any issues during runtime and the logic layer will check and report any errors found within the database and itself to the web server. Any errors found will be reported by the webserver to a system administrator. |
| Implement a notification manager into the web server to notify users of course events and administrators of system events | The web server will be able to communicate any changes in the system or courses to the relevant users. This way everyone will get the information that they need when they need it |
| Implement team structure in the database and management into the web server and logic layer | Additional tables will need to be added to the database to allow the storage of teams. As a result, the web server and the logic tier will need to be able to access, modify and leverage this team structure |

## 5: Step 6: Sketch Views and Record Design Decisions

The following refined deployment diagram includes the introduction of redundancy in the system.

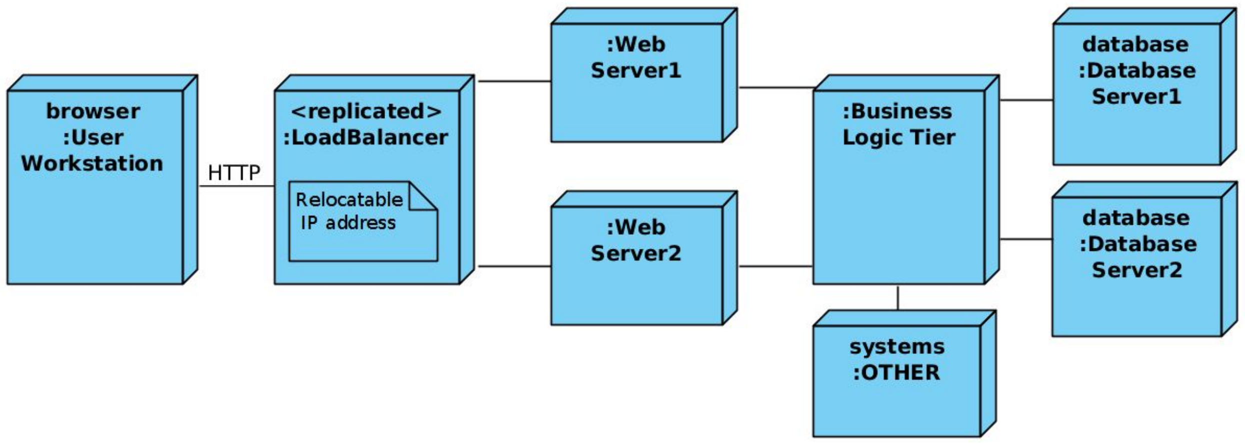
browser :User Workstation
HTTP
<replicated> :LoadBalancer
Relocatable IP address
:Web Server1
:Web Server2
:Business Logic Tier
systems :OTHER
database :Database Server1
database :Database Server2

Figure 1: Refined Deployment Diagram

The diagram represented in the above figure 1 represents the refined deployment diagram after the introduction of duplicate Web Servers as well as databases. It also includes the introduction of the load balancer. The Link to “systems :OTHER” shows where secondary school systems may be accessed.

| **Element** | **Responsibility** |
|---|---|
| Load Balancer | The load balancer dispatches, and balances the load of, requests from clients to the application servers. The load balancer also presents a unique IP address to the clients. |
| Database Server 2 | The secondary database server allows for recovery in case of Database Server 1 failure. |

## 6: Step 7: Perform Analysis of Current Design and Review Iteration Goal

The following table summarizes the design process for this iteration.

| **Not Addressed** | **Partially Addressed** | **Completely Addressed** | **Design Decisions made during the iteration** |
|---|---|---|---|
| | | UC-5 | Notification manager is being implemented into the Web Server |
| | | UC-6 | Team-based management will be added to the logic and web server, the teams will be stored in the database |

| | | | |
|---|---|---|---|
| | | UC-9 | Addition of the Tailor interface and the locate discover service allow secondary schools to easily interface with the primary school system to get the relevant data |
| | | CON-1 | Load balancer servers will allow the system to handle requests from many users simultaneously |
| | | CON-4 | Calendar export module will allow the course schedules to be exported in common calendar formats |
| | | CON-6 | Descriptive text interfaces will provide the ability for visually impaired users to access the system through screen reading software |
| | | QA-1 | The webserver and the logic layer prevent unauthorized access to the specified user |
| | | QA-3 | Load balancers maintain good performance by spreading out the requests between them during peak usage |
| | | QA-7 | Intuitive interface and team management allow lecturers to easily manage groups and send messages within them |
| | | QA-8 | This quality attribute is addressed throughout this iteration. Tactics have been put in place to ensure various accessibility requirements are met. |
| | | CRN-2 | All modules used are familiar to programming team |
| | CRN-5 | | This new architectural concern is introduced in this iteration. At this point, no relevant decisions have been made. Further iterations are needed to ensure this concern is properly addressed. |