ITERATION #2

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 fort this section: 15

Case Study: Course Management System (CMS)
ITERATION 2

Table of contents

Design Process

The following section reflects iteration 2 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 second 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: UC-2: Because it directly linked to the core features of the system. UC-5: Because of the technical issues associated with it. UC-6: Because it directly linked to the core features of the system. UC-9: Because of the technical aspects, as well as core functionality.
Quality attribute scenarios	The following quality attributes are the chose 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

Constraints	All of the constraints discussed in Assignment 2 are included as drivers.
Architectural concerns	All of the architectural concerns discussed in the above section 2.4 are included as drivers.

Iteration 2

The second iteration of the ADD design process aims to identify structures to support primary functionality.

1: Step 2: Iteration Goal By Selecting Drivers

The goal of this iteration is to address the general architecture concern of identifying structures to support primary functionality.

- UC-2: Course Management
- UC-9: Synchronize Information from secondary university systems

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

The elements that will be refined in this iteration are the database and logic layers and the web server logistics. In general, the support of functionality in this system requires the collaboration of various components associated with modules that are located on different layers.

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

In this iteration, several design concepts - in this case, architectural design patterns- are selected. From Pattern-Oriented Software Architecture, Volume 4, Appendix A, A Design Concepts Catalogue, we make the following design decisions.

Design Decision and Location	**Rationale and Assumptions**
Create a Domain Model for the application	Before we can start functional decomposition, we need an initial domain model for the system, where we identify the major entities in the domain, along with their relationships. There are no good alternatives, a domain model must be created for the system or it will appear in a nonoptimal fashion, leading to an ad hoc architecture that will be hard to maintain (≠CON-8) and hard to understand.
Identify Domain objects that map to functional requirements	Each distinct functional element of the application needs to be encapsulated in a self-contained building block- a domain object. It is possible to not consider the domain objects at this stage, but it increases the risk of possibly overlooking important requirements.
Decompose the Domain Objects into general and specialized components	Domain objects represent complete sets of functionality, but this functionality is supported by finer-grained elements located within the layers. The “components” in this pattern are what we have referred to as modules are modules.

4: Step 5: Instantiate Architectural Elements, Allocate Responsibilities, Define Interfaces

The instantiation design decisions made in this iteration are summarized in the following table.

Design Decision and Location	**Rationale and Assumptions**
Create only an initial domain model	The entities that participate in the primary use cases need to be identified and modelled but only an initial domain model is created to accelerate this phase of the design.
Map the system use cases to domain objects	Analysingallowystem's use cases allow us to identify the domain objects. To address CRN-3, we identify domain objects for all use cases presented in the ADD Introduction PDF.
Decompose the domain objects across the layers to identify layer-specific modules with an explicit interface.	This technique will ensure that modules that support all the functionalities are identified. This task is performed only on the primary uses cases. With an established set of modules, we need to test them a new architectural concern is identified here: CRN-4: Testing of a majority of the modules We only test the "majority" of the modules as modules implementing an interface can be difficult to implement an independent test.

While the structures and interfaces are identified in the above table, they are captured in the next step.

5: Step 6: Sketch Views and Record Design Decisions

As a result of all the decisions in step 5 of iteration 2, multiple diagrams were created.

- Figure 1 shows an initial domain model for the system
- Figure 2 shows the domain objects instantiated for the system.
- Figure 3 is a module view with modules derived from our use cases. Explicit interfaces are not shown but their existence is assumed.

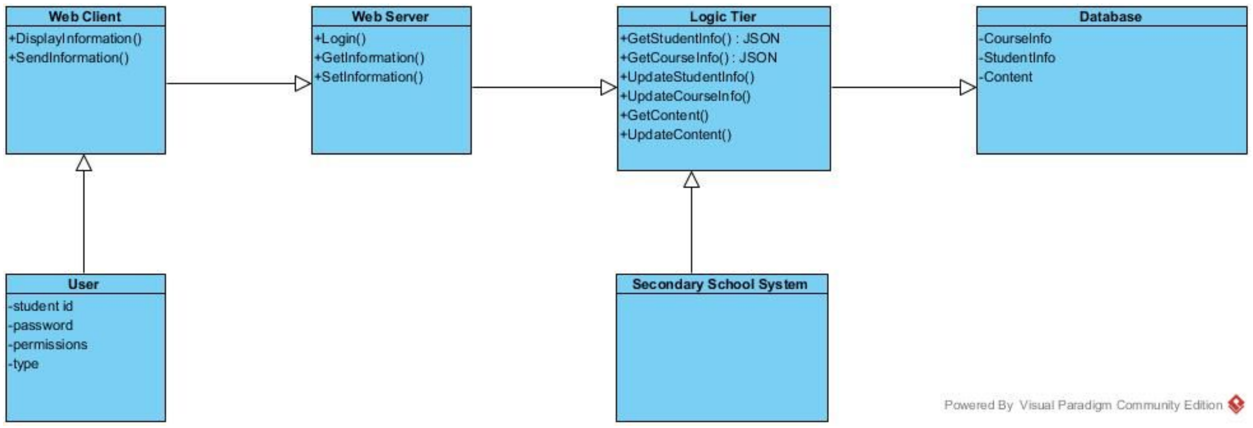

Figure 1: Initial Domain Model

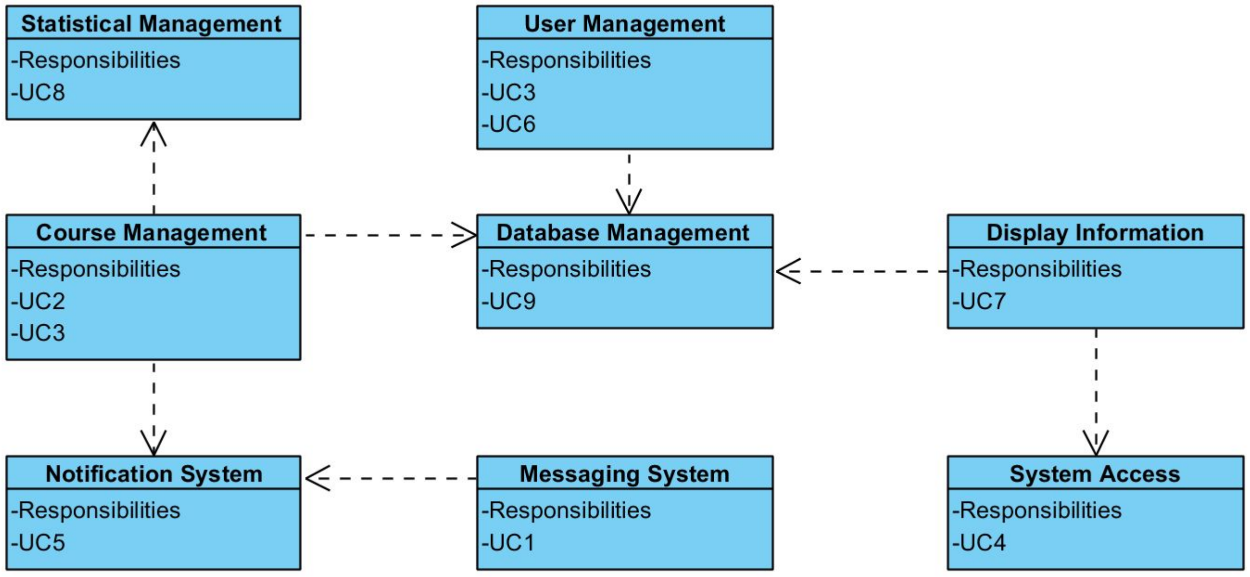

Figure 2: Domain Objects Associated with Use Cases

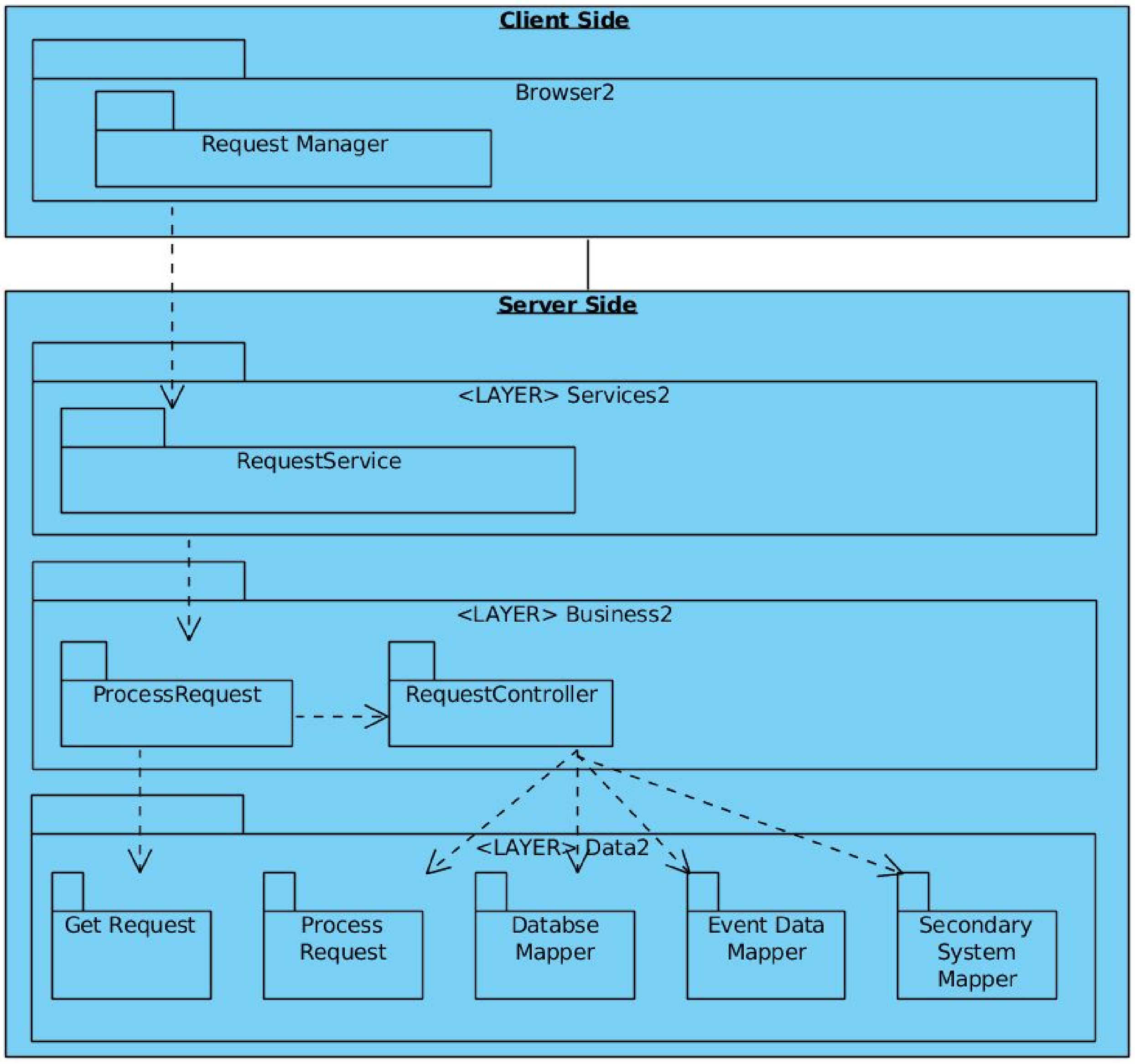

Figure 3: Extended Module View

The responsibilities of the elements in the above Extended Module View (Figure 3) are summarized in the following table:

Element	**Responsibility**
Request Manager	Gets request from users via the browser
Request Service	Requests a data service from the user. Or fulfils a page change via the browser.
Process Request	Process request from the user
Request Controller	Responsible for business logic related to management of requests and

	Events.
Database Mapper	Responsible for persistence operations related to the events
EventData Mapper	Responsible for persistence operations related to the events
Secondary System Mapper	Responsible for persistence operations related to the events

USE-CASE 2: Course Management

The next figure shows the initial sequence diagram for UC-2. It shows how a user would access all and any content on a specific page. Upon launch, when a user asks to view content, the content is requested from an actor via the browser. Then the browser connects to the web server who processes the requests and sends a request to the database where the request is processed and the database finds and sends the file back to the user to the web server who then processes the file and displays configuration to the user when given the proper permission rights.

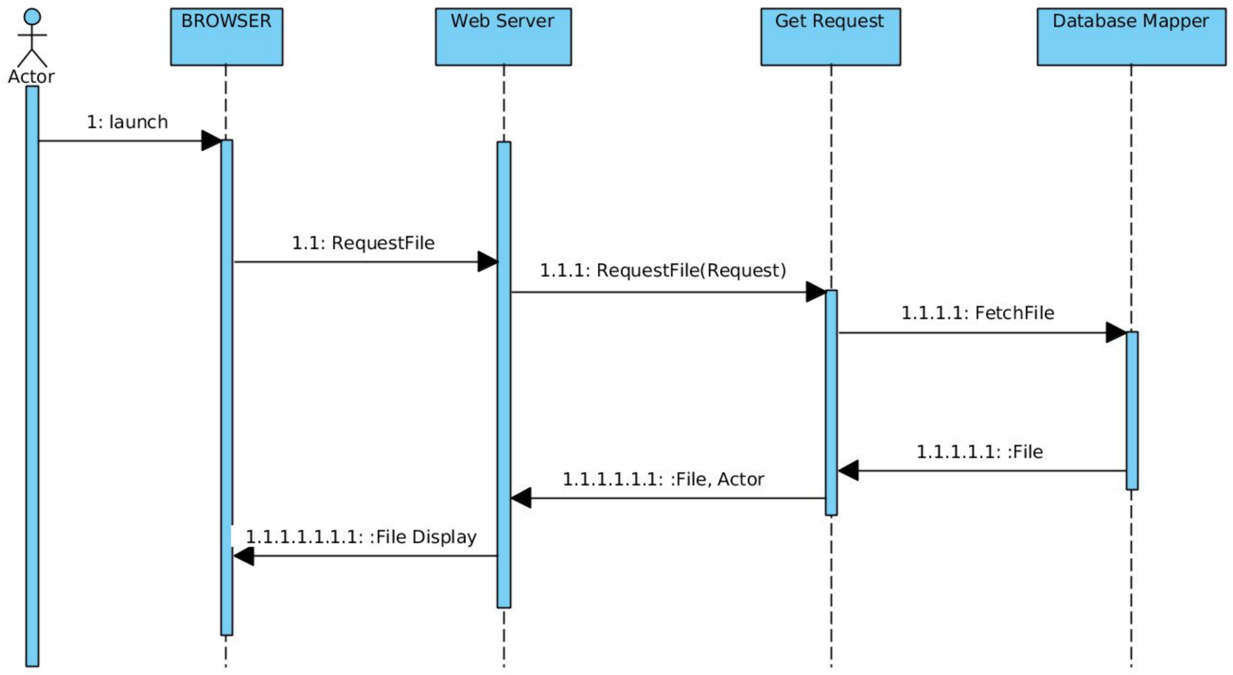

Figure 4: Sequence Diagram for UC-2

USE-CASE 9: Synchronize Information

The next figure shows the initial sequence diagram for UC-9. It shows how a user would access the secondary university system via the secondary school client. This example shows the application accessing a primary school information schema via the secondary school client. The user connects to the Secondary School Client, which then connects to the secondary school web server. This activates the logic tier which then gets the information from the Primary School Database. The data is then sent back to the actor.

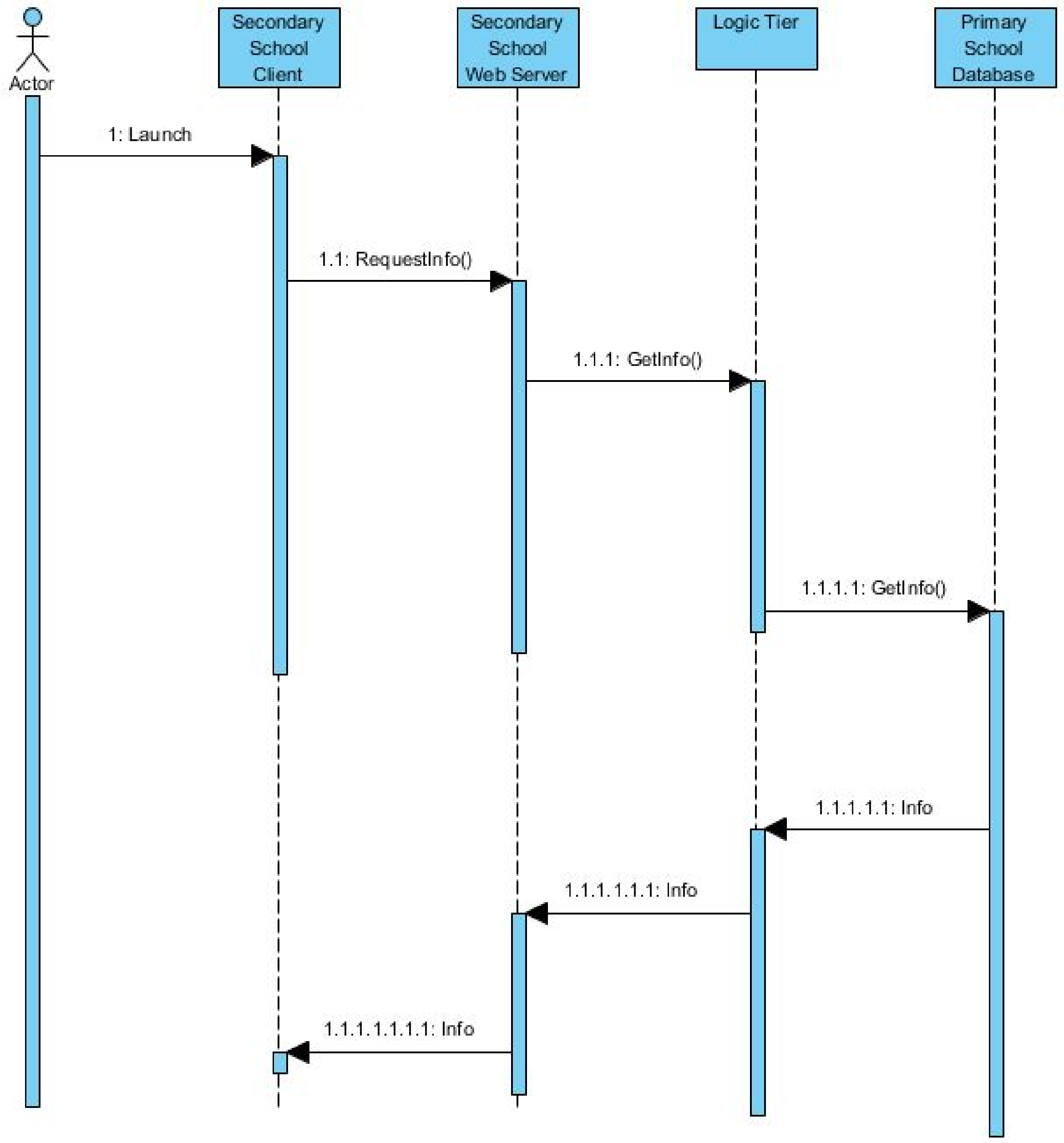

Figure 5: Sequence Diagram for UC-9

From the interactions identified in the above sequence diagrams, initial methods for interfaces of the interfacing elements can be identified.

Element	Description
Browser	The browser acts as an intermediate between the actor and the system.
Web Server	The web server allows for communication between the Client and the database
Request Manager	Manages File and Information Requests from the User and Server
Database Mapper	The database mapper allows for communication between the web server and the database
Client	Secondary users that wish to communicate information across services via the Web Logic Tier
Logic Tier	The logic tier allows for different levels of access to be provided to the database based on whether the requester is part of the primary or a secondary school system

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

The decisions made in this iteration provided an initial understanding of how functionality is supported through the system. Primary use case modules were identified as well as modules associated with other components.

Also part of this iteration we have identified a new architectural concern, which was added to the kanban board below. Drivers completely addressed in iteration 1 are now removed.

The following table summarizes the design progress.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions made during the iteration
		UC2	Modules across the layers and preliminary interfaces to support this use case have been identified.
	UC-5		Modules across the layers and preliminary interfaces to support this use case have been found but not addressed.
	UC-6		Modules across the layers and preliminary interfaces to support this use case have been found but not addressed.
		UC-9	Modules across the layers and preliminary interfaces to support this use case have been identified.
	CON-1		No relevant decisions were made.
		CON-3	Modules responsible for connecting to an existing relational database are now in place.

	CON-4		No relevant decisions were made for export functionalities in this iteration.
		CON-5	Modules for accessing external data were identified in this iteration via UC-9.
CON-6			No relevant design decisions were made to UI in this iteration
		CON-7	The Business logic tier of the system provides read-only access to secondary schools
		CON-8	Components have been defined which are easy to maintain and test
	QA-1		Each user has permissions and can only make changes within those permissions. Decisions regarding logging have not been made during this iteration
	QA-2		No relevant design decisions were made in this iteration
		QA-5	Secondary schools will have access to the database of the primary school through the use of the logic tier API layer which will allow access only
QA-7			No relevant design decisions have been made so far regarding UI
QA-8			No relevant design decisions have been made so far regarding UI
	CRN-2		Additional technologies are identified using the team's knowledge.
		CRN-3	Modules associated with all the uses cases have been identified and a work assignment matrix is now easily created given a team.
		CRN-4	The architectural concern of unit testing modules, introduced in this iteration, is partially solved through the use of an inversion of control approach to connecting the components associated with the modules.