

ADD Step 1: Review Inputs

Category	Details																					
Design Process	This is a greenfield system from a mature domain. The purpose is to produce a sufficiently detailed design to support the construction of the system.																					
Primary Functional Requirements	<p>From the use cases presented from the first deliverable, the primary ones were determined to be:</p> <ul style="list-style-type: none">• UC-1 Monitor System Status: Essential Component of CMS• UC-5 Course Portal Display: Essential Component of CMS• UC-6 User Profile Utilities: Essential Component of CMS• UC-7 Manage Database: Essential Component of CMS• UC-10 User Login: Essential Component of CMS• UC-11 User Management: Essential Component of CMS																					
Quality Attribute Scenarios	<table><tr><th>Scenario ID</th><th>Importance to Stakeholders</th><th>Difficulty of Implementation according to architect</th></tr><tr><td>QA-1</td><td>High</td><td>High</td></tr><tr><td>QA-2</td><td>Medium</td><td>Medium</td></tr><tr><td>QA-3</td><td>High</td><td>High</td></tr><tr><td>QA-4</td><td>High</td><td>High</td></tr><tr><td>QA-5</td><td>High</td><td>Medium</td></tr><tr><td>QA-6</td><td>High</td><td>Medium</td></tr></table> <p>Only QA-1, QA-3, QA-4, QA-5, QA-6 are selected as drivers</p>	Scenario ID	Importance to Stakeholders	Difficulty of Implementation according to architect	QA-1	High	High	QA-2	Medium	Medium	QA-3	High	High	QA-4	High	High	QA-5	High	Medium	QA-6	High	Medium
Scenario ID	Importance to Stakeholders	Difficulty of Implementation according to architect																				
QA-1	High	High																				
QA-2	Medium	Medium																				
QA-3	High	High																				
QA-4	High	High																				
QA-5	High	Medium																				
QA-6	High	Medium																				
Constraints	All the constraints discussed in the first deliverable are																					

	included as drivers.
Architectural Concerns	All the architectural concerns discussed in first deliverable are included as drivers.

Iteration 3: Addressing Quality Attribute Scenario Driver (QA-3)

This section presents the results of the activities that are performed in each of the steps of ADD in the third iteration of the design process. Building on the fundamental structural decisions made in iterations 1 and 2, we can now start to reason about the fulfillment of some of the more important quality attributes. This iteration focuses on just one of these quality attribute scenarios

Step 2: Establish Iteration Goal by Selecting Drivers

For iteration 3, the architect focused on QA-3 quality attribute scenario. A failure occurs in the CMS system during an operation. The CMS system will need to shut down and restart immediately.

Step 3: Choose One or More Elements of the System to Refine

For this particular scenario, the elements that need to be refined are:

- Application Server
- Database Server

Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers

Design Decisions and Location	Rationale and Assumption
Introduce active redundancy, by replication application server and other critical components such as database.	When replicating these critical elements, the CMS system can withstand the failure of one of the replicated elements without affecting functionality.
Backup database regularly	In case the database server crashes, due to malfunction or such. Having backup data

	will ensure previous data, and prevent data loss.
Introduce element from message queue technology family	Traps received from time servers are placed in the message queue and then retrieved by the application. Use of a queue will guarantee that traps are processes and delivered in order (QA-1)

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

The instantiation design decisions are summarized in the following table:

Design Decisions and Location	Rationale
Deploy Message queue on separate node	Deploying the message queue on separate node will guarantee that no traps are lost in case of application failure. This node is replicated using the tactic of active redundancy, but only one copy receives and treats events coming from the CMS system.
Use active redundancy and load balancing in the application server	Load balancer needs to be deployed to regulate traffic for multiple application server. This will introduce a new architectural concern CRN-5: Manage states in replicas.
Implementing load balancing and redundancy using support	There are many technological options for load balancing and redundancy that can be implemented without having to develop ad hoc solution that is harder to support and less mature

Step 6: Sketch Views and Record Design Decisions

Figure 1 shows the refined deployment diagram that includes the introduction of redundancy in the system.

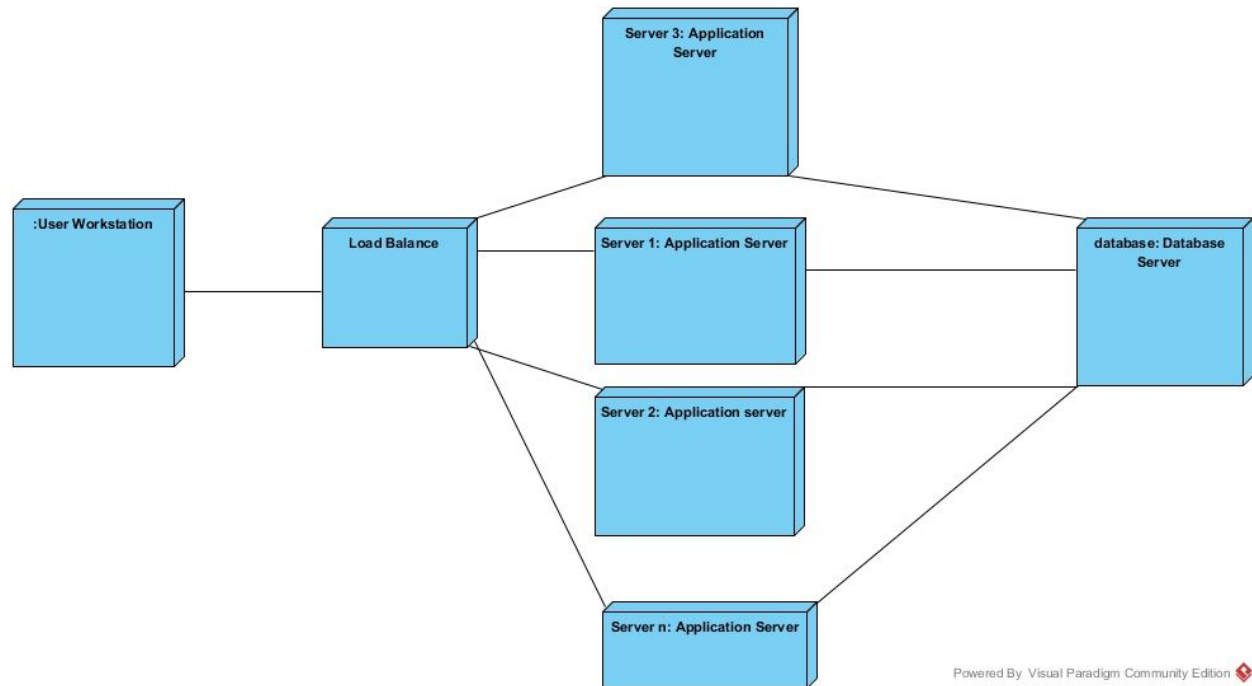


Figure 1: Deployment Diagram

Element	Responsibility
Load Balance	Takes the requests coming from users to application servers.

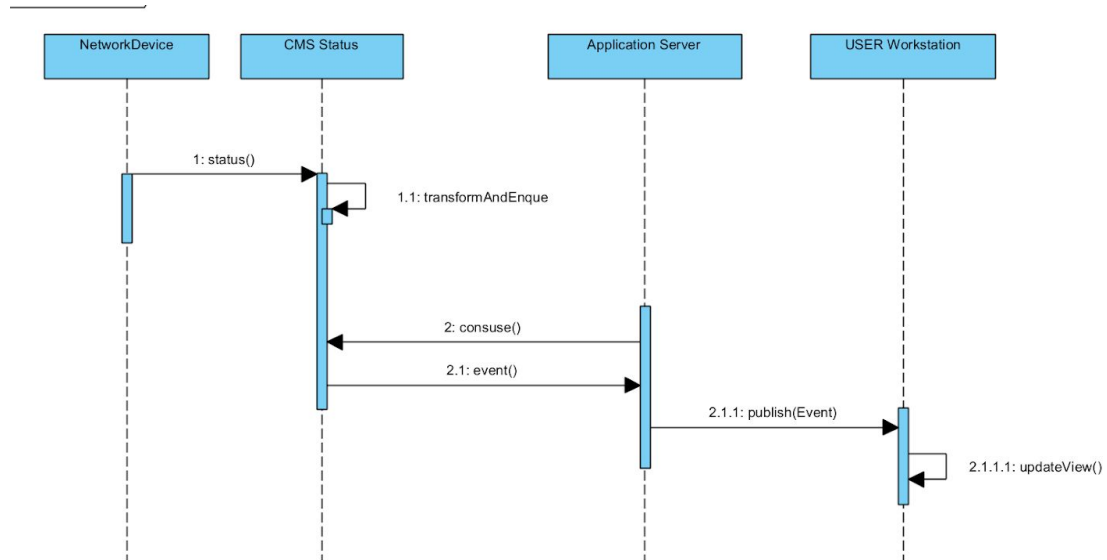


Figure 2: Diagram

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