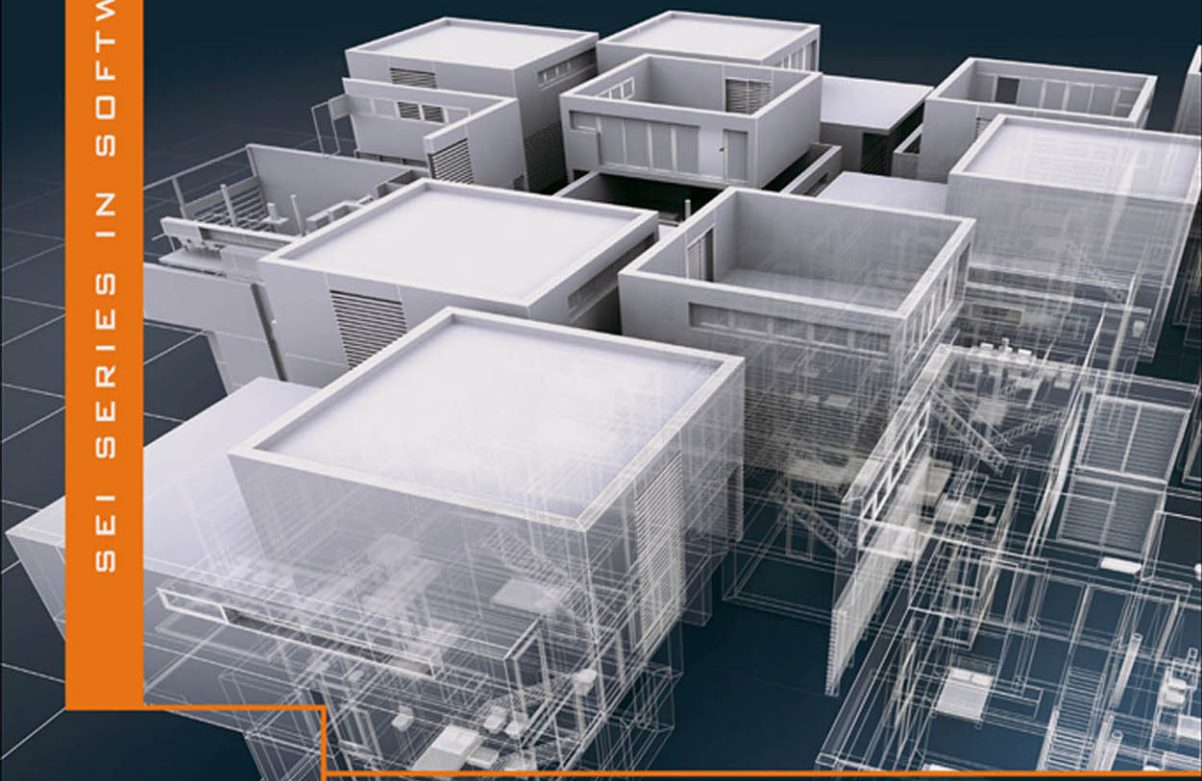
SEI SERIES IN SOFTWARE ENGINEERING

Designing Software Architectures

A Practical Approach

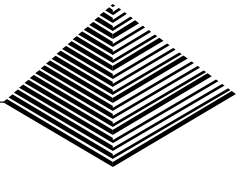


Humberto Cervantes

Rick Kazman

Designing Software Architectures

4



Case Study: FCAPS System

We now present a case study of using ADD 3.0 for a greenfield system in a mature domain. This case study details an initial design round composed of three iterations and is based on a real-world example. We first present the business context, and then we summarize the requirements for the system. This is followed by a step-by-step summary of the activities that are performed during the ADD iterations.

4.1 Business Case

In 2006, a large telecommunications company wanted to expand its Internet Protocol (IP) network to support "carrier-class services", and more specifically high-quality voice over IP (VOIP) systems. One important aspect to achieve this goal was synchronization of the VOIP servers and other equipment. Poor synchronization results in low quality of service (QoS), degraded performance, and unhappy customers. To achieve the required level of synchronization, the company wanted to deploy a network of time servers that support the Network Time Protocol (NTP). Time servers are formed into groups that typically correspond to geographical regions. Within these regions, time servers are organized hierarchically in levels or *strata,* where time servers placed in the upper level of the

hierarchy (stratum 1) are equipped with hardware (e.g., Cesium Oscillator, GPS signal) that provides precise time. Time servers that are lower in the hierarchy use NTP to request time from servers in the upper levels or from their peers.

Many pieces of equipment depend on the time provided by time servers in the network, so one priority for the company was to correct any problems that occur on the time servers. Such problems may require dispatching a technician to perform physical maintenance on the time servers, such as rebooting. Another priority for the company was to collect data from the time servers to monitor the performance of the synchronization framework.

In the initial deployment plans, the company wanted to field 100 time servers of a particular model. Besides NTP, time servers support the Simple Network Management Protocol (SNMP), which provides three basic operations:

- `set()` operations: change configuration variables (e.g., connected peers)
- `get()` operations: retrieve configuration variables or performance data
- `trap()` operations: notifications of exceptional events such as the loss or restoration of the GPS signal or changes in the time reference

To achieve the company's goals, a management system for the time servers needed to be developed. This system needed to conform to the FCAPS model, which is a standard model for network management. The letters in the acronym stand for:

- *Fault management.* The goal of fault management is to recognize, isolate, correct, and log faults that occur in the network. In this case, these faults correspond to traps generated by time servers or other problems such as loss of communication between the management system and the time servers.
- *Configuration management.* This includes gathering and storing configurations from network devices, thereby simplifying the configuration of devices and tracking changes that are made to device configurations. In this system, besides changing individual configuration variables, it is necessary to be able to deploy a specific configuration to several time servers.
- *Accounting.* The goal here is to gather device information. In this context, this includes tracking device hardware and firmware versions, hardware equipment, and other components of the system.
- *Performance management.* This category focuses on determining the efficiency of the current network. By collecting and analyzing performance data, the network health can be monitored. In this case, delay, offset, and jitter measures are collected from the time servers.
- *Security management.* This is the process of controlling access to assets in the network. In this case, there are two important types of users: technicians and administrators. Technicians can visualize trap information and configurations but cannot make changes; administrators are technicians who can visualize the same information but can also make changes to configurations, including adding and removing time servers from the network.

Once the initial network was deployed, the company planned to extend it by adding time servers from newer models that might potentially support management protocols other than SNMP.

The remainder of this chapter describes a design for this system, created using ADD 3.0.

4.2 System Requirements

Requirement elicitation activities had been previously performed, and the following is a summary of the most relevant requirements collected.

4.2.1 Use Case Model

The use case model in Figure 4.1 presents the most relevant use cases that support the FCAPS model in the system. Other use cases are not shown.

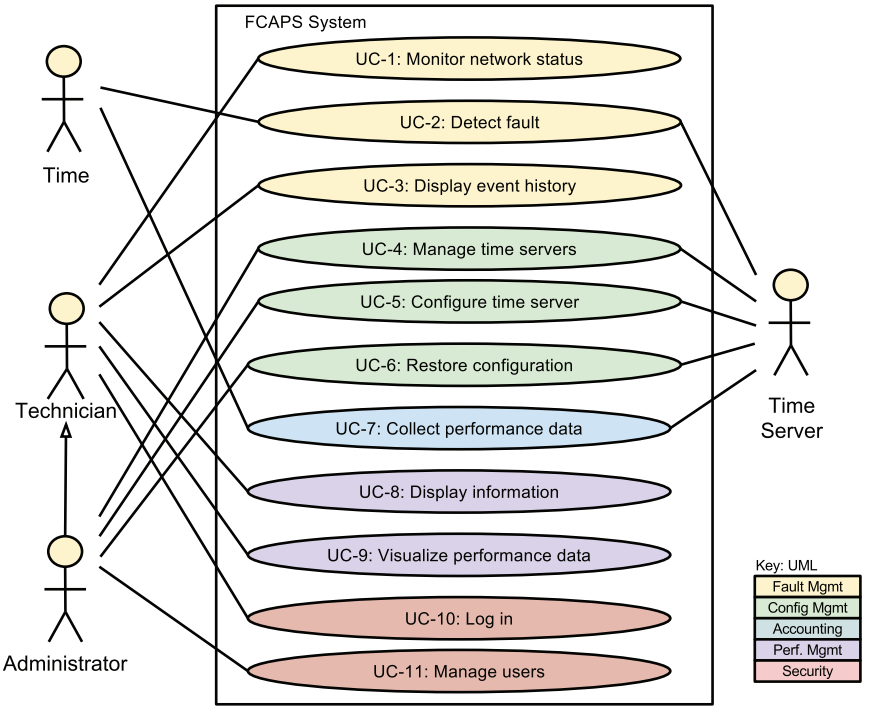


FIGURE 4.1 Use case model for the FCAPS system

Each of these use cases is described in the following table:

Use Case	Description
UC-1: Monitor network status	A user monitors the time servers in a hierarchical representation of the whole network. Problematic devices are highlighted, along with the logical regions where they are grouped. The user can expand and collapse the network representation. This representation is updated continuously as faults are detected or repaired.
UC-2: Detect fault	Periodically the management system contacts the time servers to see if they are "alive". If a time server does not respond, or if a trap that signals a problem or a return to a normal state of operation is received, the event is stored and the network representation observed by the users is updated accordingly.
UC-3: Display event history	Stored events associated with a particular time server or group of time servers are displayed. These can be filtered by various criteria such as type or severity.
UC-4: Manage time servers	The administrator adds a time server to, or removes a time server from, the network.
UC-5: Configure time server	An administrator changes configuration parameters associated with a particular time server. The parameters are sent to the device and are also stored locally.
UC-6: Restore configuration	A locally stored configuration is sent to one or more time servers.
UC-7: Collect performance data	Network performance data (delay, offset, and jitter) is collected periodically from the time servers.
UC-8: Display information	The user displays stored information about the time server—configuration values and other parameters such as the server name.
UC-9: Visualize performance data	The user displays network performance measures (delay, offset, jitter) in a graphical way to view and analyze network performance.
UC-10: Log in	A user logs into the system through a login/password screen. Upon successful login, the user is presented with different options according to their role.
U-11: Manage users	The administrator adds or removes a user or modifies user permissions.

4.2.2 Quality Attribute Scenarios

In addition to these use cases, a number of quality attribute scenarios were elicited and documented. The six most relevant ones are presented in the following table. For each scenario, we also identify the use case that it is associated with.

ID	Quality Attribute	Scenario	Associated Use Case
QA-1	Performance	Several time servers send traps to the management system at peak load; 100% of the traps are successfully processed and stored.	UC-2
QA-2	Modifiability	A new time server management protocol is introduced to the system as part of an update. The protocol is added successfully without any changes to the core components of the system.	UC-5
QA-3	Availability	A failure occurs in the management system during normal operation. The management system resumes operation in less than 30 seconds.	All
QA-4	Performance	The management system collects performance data from a time server during peak load. The management system collects all performance data within 5 minutes, while processing all user requests, to ensure no loss of data due to CON-5.	UC-7
QA-5	Performance, usability	A user displays the event history of a particular time server during normal operation. The list of events from the last 24 hours is displayed within 1 second.	UC-3
QA-6	Security	A user performs a change in the system during normal operation. It is possible to know who performed the operation and when it was performed 100% of the time.	All

4.2.3 Constraints

Finally, a set of constraints on the system and its implementation were collected. These are presented in the following table.

ID	Constraint
CON-1	A minimum of 50 simultaneous users must be supported.
CON-2	The system must be accessed through a web browser (Chrome V3.0+, Firefox V4+, IE8+) in different platforms: Windows, OSX, and Linux.
CON-3	An existing relational database server must be used. This server cannot be used for other purposes than hosting the database.
CON-4	The network connection to user workstations can have low bandwidth but is generally reliable.
CON-5	Performance data needs to be collected in intervals of no more than 5 minutes, as higher intervals result in time servers discarding data.
CON-6	Events from the last 30 days must be stored.

4.2.4 Architectural Concerns

Given that this is greenfield development, only a few general architectural concerns are identified initially, as shown in the following table.

ID	Concern
CRN-1	Establishing an overall initial system structure.
CRN-2	Leverage the team's knowledge about Java technologies, including Spring, JSF, Swing, Hibernate, Java Web Start and JMS frameworks, and the Java language.
CRN-3	Allocate work to members of the development team.

Given these sets of inputs, we are now ready to proceed to describe the design process, as described in Section 3.2. In this chapter, we present only the final results of the requirements collection process. The job of collecting these requirements is nontrivial, but is beyond the scope of this chapter.

4.3 The Design Process

We now ready to make the leap from the world of requirements and business concerns to the world of design. This is perhaps the most important job for an architect—translating requirements into design decisions. Of course, many other decisions and duties are important, but this is the core of what it means to be an architect: making design decisions with far-reaching consequences.

4.3.1 ADD Step 1: Review Inputs

The first step of the ADD method involves reviewing the inputs and identifying which requirements will be considered as drivers (i.e., which will be included in the design backlog). The inputs are summarized in the following table.

Category	Details
Design purpose	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	From the use cases presented in Section 4.2.1, the primary ones were determined to be: UC-1: Because it directly supports the core business UC-2: Because it directly supports the core business UC-7: Because of the technical issues associated with it (see QA-4)