CSCE 742 - Project Part 3 - Architectural Description

Due Date: Sunday, November 18th, 11:59 PM (in PDF format, via Dropbox)

Your job, this semester, is to understand, document, and extend the architecture of a non-trivial real-world software system. In the last phase of the project, you established system scope, context, and requirements. In this phase, you will describe the architecture at multiple layers of abstraction and also from multiple viewpoints.

In this assignment, you will examine the system at **two** levels of granularity:

- The top-level system itself.
- One “architecturally interesting” subsystem to examine in more detail.

For both the top-level system and the chosen subsystem, you should describe the **functional viewpoint** and **one additional viewpoint** of your choosing. **This additional viewpoint cannot be the context view**, as we have explored the system context in Part 2 of the project.

At least two important perspectives, as identified by the previous phase, should be presented, following the descriptions of the views for the top-level and subsystem chosen. In describing the perspectives, use analysis techniques from Rozanski & Woods and Bass, Clements, & Kazman that are relevant to the system that you are describing. The perspectives should be a vehicle for critiquing the architecture in terms of how well it meets its quality goals, as you see them.

Each view should include both an appropriate visualization, as well as textual descriptions. For the functional view, you should include UML sequence diagrams for at least the three most important scenarios for the system/subsystem being modeled. For the top-level model, these may (or may not) correspond to use cases, but the dynamics of important use-cases should be derivable from the sequence diagrams. The entities in the sequence diagrams should correspond to architectural elements - they do not have to be at the individual class level.

For the top-level system, the idea is to trace the views to the goals and scenarios that you proposed in the previous part of the project. The alternate viewpoint should be of ‘high’ importance, as ranked from the previous phase.

For the subsystem, you are free to choose any viewpoint that you think would be interesting. If examining an immediate subsystem of the top-level architecture, then the interface boundaries of the subsystem should obviously match the subsystem interface in the top level. If choosing a nested subsystem, please explain where the subsystem ‘lives’ in the context of the larger system.

Then, based on the goals and scenarios, for either the top-level system or the chosen subsystem, propose one of (1) an alternate architecture (refactoring) that would better support the goals of the system, or (2), an extension of the architecture based on the change cases described in the previous phase of the project.

Your submission should be organized as follows (**bolded items are new**):

1. Introduction
2. System Overview
    a. Purpose and Scope
    b. Audience
    c. Why the System is Architecturally Interesting
    d. Quality Properties
    e. Architectural Design Approach
3. **Glossary of Terms**
4. Stakeholders and Requirements
    a. Stakeholders
    b. Overview of Requirements
    c. System Scenarios
    d. Change Cases
5. **Architectural Focus**
    a. Goals
    b. Constraints
    c. Architectural Principles
6. Architectural Views for Top-Level System
    a. Context View
    b. Functional View
    c. View of Your Choosing
7. **Architectural Views for Chosen Subsystem**
    a. Functional View
    b. View of Your Choosing
8. **Decisions and Alternatives (put your proposed extensions here)**

Guidance

First, for perspectives, I would like you to integrate them into Sections 6 and 7. In Section 6, describe how perspectives affect the top-level system; similarly, describe how perspectives affect the subsystem in Section 7.

In the Architectural Design Approach section, I'd like you to describe the approach you have used to describe the architecture; essentially what you are doing in this class. For example, the following is an excerpt from what one team describing the Android operating system wrote:

In analyzing the architectural design of Android, due to its incredible size, we were forced to focus our analysis on just a small part of the overall system. In this document we focus on the design of the Intent system, which governs how applications are allowed to interact with one another, and AppWidgets, which utilize the Intent system in their operation.

We theorized on what qualities would be important for the system, then determined how the design of the system accommodated those qualities. We examined existing documentation for the Intent and AppWidgets components that is freely available on the Internet. We also downloaded the source code as it exists today and imported it into Eclipse in order to examine the code itself. Using ObjectAid, an extension to Eclipse, we were able to generate class diagrams that provided further insights in how the classes interacted with one another.

As a result, we were able to piece together how the Intent system provides a secure method for activities within Android to communicate with each other and share resources that is also easy for a developer to implement and will provide for extensibility.

Diagrams must always be supported with explanatory text. Even if you are later describing each of the elements (say, the components in a component in the component diagram), you need to have prose related to the “whole picture”. Why is the diagram included? How do the elements of the diagram work together to solve some problem? For context diagrams, please describe the external entities.

Strive to make your views consistent: if you describe a component in your concurrency diagram for a system / subsystem, it probably should exist in your functional diagram.

Requirements must be specific and verifiable. If you can’t think of a test case that can determine whether or not the requirement holds for the system, you are probably writing a goal, not a requirement.

Consider the essential aspects of your architecture in your description. For example, if you are describing the information view of a object/relational mapper, it should describe how tables are constructed from objects.

System dynamics must be considered. You should have UML sequence diagrams for at least the most important 3-4 scenarios for the system/subsystem being modeled. For the top-level model, these may (or may not) correspond to use cases, but the dynamics of important use-cases should be derivable from the sequence diagrams.

Any text or figures copied or paraphrased from external sources must be given proper citations. We expect you to properly attribute your work. Failure to do so would be academic dishonesty, as covered in the syllabus.

The purpose and scope section should define the scope of what is being documented. In your case, you should summarize that this is not a complete description of the architecture, but a slice that contains two architectural layers, two views for each layer and 3-4 perspectives. Note that the scope here does not mean "the scope of the system" in the sense of a context diagram; this comes later in Section 6.1. A reasonable section would be something like:

The purpose is to understand, document, and eventually extend the architecture of project YYY. Project YYY has a non-trivial architecture because (information from your Phase I writeup here). In this document, we are not attempting to describe all aspects of the architecture, but the most architecturally interesting information. We will describe this information using Rozanski and Woods idea of viewpoints and perspectives, and focus on two architectural layers of the system. Part of this document (the functional view) will include the interfaces of the significant architectural components and their interconnection.

The Architectural Design Approach section should describe and justify the layers, viewpoints, and perspectives used.

The extension/refactoring proposal should contain a page at most; ideally, it would be derived from one of the change cases from Phase II with slightly more information as to why it is important to the system and architecturally interesting.