Documenting an Architecture

Software Architecture

Semester 1, 2022

Richard Thomas & Brae Webb

Summary

In this assignment, you will demonstrate your ability to *understand* and subsequently *communicate* the architecture of an existing software project.

1. First, you need to choose a suitable open source software project. The project must have non-trivial functionality and architecture.
2. You will write a report which describes the architecture of your selected software project.

1 Introduction

The digital world relies heavily on open source software, as seen by the recent log4j vulnerability.[1] Fortunately, open source developers often maintain high quality documentation for the users of their projects. Unfortunately however, many open source projects do not maintain the same high quality documentation for the architecture of their software projects. This can cause difficulty for developers who want to contribute to the project, but first need to understand it.

In this assignment, you have the chance to right this wrong. Your task is to find an open source software project with a sufficiently complex architecture and document it. You may optionally choose to share this documentation with the project developers. You are encouraged to do this, as the perspective of a newcomer to a project is often invaluable to the seasoned developers.

Before looking for projects, read some of the architecture documentation written by students at TU Delft: https://delftswa.gitbooks.io/desosa2016/content and https://delftswa.github.io.

It would also be advantageous to read through one of the architecture descriptions in either volume of The Architecture of Open Source Applications: http://aosabook.org.

2 Finding a Project

Criteria for the software project:

- The project cannot be covered in the tutorials or by the TU Delft students above.
- The project must have at least one release within the last year.

Places to look for projects:

- GitHub explore page: https://github.com/explore;
- Apache project list: https://apache.org/index.html#projects-list;
- Awesome Open Source https://awesomeopensource.com/;
- in class discussion with other students;
- or, ask your tutor.

[1] https://www.cisa.gov/uscert/apache-log4j-vulnerability-guidance

3 Report Structure

Title Name of the software project.

Abstract Summarise the key points of your document.

Introduction Describe the software project, explaining its key functionality and target users.

ASRs Describe the project's architecturally significant requirements.

Context Provide an overview of the software system's context and its external dependencies.

Architecture Describe the software's architecture in detail.

Critique Evaluate the software's architecture, describing its advantages and disadvantages.

Conclusion Highlight the key points or lessons learnt about the software's architecture.

4 Report Content

How you present the information in your report, is up to you. You need to select an appropriate notation to provide a visual representation of the software's architecture. You need to select appropriate ways of describing the architecture.

The "Architectural Views"[2] notes describe different types of views that can be used to describe aspects of a software architecture. The notes also describe two different notations that can be used to describe parts of a software architecture. You need to decide which views are appropriate to provide an accurate, informative, and complete description of the project's architecture. The notation you use to provide visual information about the architecture must convey that information appropriately. Your textual commentary needs to expand on the information provided in the diagrams.

The introduction section of your report **must** include a link to the project's main repository (e.g. Ruby on Rails Repo[3]) and, if useful, any other important sites related to the project (e.g. Ruby on Rails[4]). The introduction should provide enough information so that a reader who is not familiar with the project can understand what it delivers.

You need to identify what you think are the project's architecturally significant requirements (ASRs) and, in particular, the quality attributes that are important to the project's success. You need to justify why these requirements are important. You should consider how these attributes would be prioritised to make decisions when trade offs need to be made in the design. Your critique should describe how well the software's architecture supports delivering the ASRs that you identified as being important.

The context should help the reader to understand who uses the system and its dependencies on other systems. It should elaborate on the general introduction of the system to provide a starting basis to understand the software system's architecture.

Provide a detailed description of the software system's architecture. It should cover all important aspects of the architecture. You should include diagrams to help the reader understand the details of the architecture. You should describe any security risks inherent in the software architecture. Your critique should evaluate and discuss what security design principles appear to have been followed in the design of the software and how well they guard against the security risks. Your critique should also assess the advantages and potential drawbacks of the architectural design.

You are not expected to get down to the level of describing the detailed design of the software. You should not need to provide class diagrams for the entire system. You may need to provide class diagrams

[2] https://csse6400.uqcloud.net/schedule/#week2
[3] https://github.com/rails/rails
[4] https://rubyonrails.org/

to highlight important features supported by the architecture. For example, a class diagram showing a plug-in API, and how the application uses a plug-in, would be informative.

You may use tools to generate diagrams and to extract information from source code that is used to create diagrams or other documentation artefacts.

The audience of your report is software developers who are familiar with software architecture and design. You may assume that the reader is familiar with UML and C4 notations.

5 Presentation

The presentation assessment task scheduled later in the semester will be based on the project you select for this assignment. In the presentation you will provide a more in-depth critique of the project's architecture, based on your further experience in this course.

6 Submission

Your report document is due at **16:00 (AEST) on 4th April 2022**. Late submissions will **not** be marked. The word limit for your report is a maximum of 2200 words, including bibliography. Images are not part of the word count, but text in labels will be counted. TurnItIn's word count will be taken as the official number of words in your report. Markers will not consider content past the 2200 word limit when grading your report.

There is no minimum word limit for your report. Conciseness is appreciated by markers, and being able to deliver information concisely is an important professional skill. You may use bullet points, rather than descriptive paragraphs, if it helps you deliver information clearly and concisely.

You will submit your report as a **PDF** file to a TurnItIn link on BlackBoard. The textual content of your report **must** be saved as text within the PDF document. If the text is converted to images, your report will **not** be marked. TurnItIn may reject a PDF file where all text is converted to images, or TurnItIn may accept the file but the word count will be very low or zero. It is your responsibility to ensure that your submitted PDF file meets all submission requirements. You may submit your report multiple times before the submission deadline. A late submission due to problems uploading the file will **not** be marked.

The content of your report, including any quoted material, must be in English.

7 Criteria

You will be assessed on how well you

- describe the system's context and target users,
- justify the ASRs you identify,
- describe the system's architecture,
- assess security risks inherent in the architecture, and
- critique the system's architecture.

A detailed marking rubric is available via the TurnItIn link on BlackBoard.

8 Academic Integrity

As this is a higher-level course, you are expected to be familiar with the importance of academic integrity in general, and the details of UQ's rules. If you need a reminder, review the Academic Integrity Modules[5]. Submissions will be checked to ensure that the work submitted is not plagiarised. If you have quoted or paraphrased any material from another source, it must be correctly cited and referenced[6]. Use the IEEE referencing style[7] for citations and your bibliography.

Uncited or unreferenced material will be treated as not being your own work. Extensive quotation or minor rephrasing of material from external sources should be avoided. Large blocks of cited material from other sources, even if paraphrased, will be considered to be of no academic merit. In all cases, any material that you cite must support the arguments and points that you are making in your document.

[5] https://web.library.uq.edu.au/library-services/it/learnuq-blackboard-help/academic-integrity-modules

[6] https://web.library.uq.edu.au/node/4221/2

[7] https://libraryguides.vu.edu.au/ieeereferencing/gettingstarted