Research Project
Exploring architectural design decisions in issue tracking systems

Filipe Alexandre Rosa Capela
S4040112

April 2020

Contents

1 Introduction

This research project is inserted in the context of an internship for the course INMSTAG-08.2019-2020.2B lectured at the University of Groningen, where myself, Filipe Alexandre Rosa Capela, will be under the supervision of dr. Mohamed Soliman and prof. Paris Avgeriou.
In this project I will be conducting an experiment to extract whether certain changes to the source code of projects which utilize issue tracking systems can relate to changes in the architecture of their software systems.
In early meetings discussing the possible directions that could be followed to draw the desired conclusions, three possibilities arose:

1. To capture changes in the components structure, using the Arcan tool. The challenges that I could possibly encounter were:
    - To understand which of the changes in the dependency graph are architectural decisions.
    - How to compare the dependency graphs between different versions.
    - The need for a high performance machine since thousands of graphs would be generated, and hence the need to create very well designed algorithms.
2. Extracting design patterns and architectural tactics using the Pinot [1] and Archie [2] tools. The challenges that I could possibly encounter were:
    - Create a process that could iterate through all the commits and run the tools for each commit.
    - The fact that Pinot and Archie operate differently (one on the CLI and another through Eclipse) makes it hard to combine the outputs.
3. Detecting changes in the technology stack used throughout a projects lifecycle. The challenges that I could possibly encounter were:
    - How to develop a process capable of detecting these changes. Since different projects use different build tools (Maven, Gradle, Ant, etc...)
    - The same as in direction #2, there needs to be a way to loop through all the commits in a GitHub repository.

After some thought about which direction to adopt, I decided to go with the **second approach**, to use Pinot and Archie in order to detect Design and Architectural Patterns. I took this decision since it seems the most suitable given the courses I attended in the University of Groningen taught by the same two lecturers that are supervising me in this internship. Another reason is the fact that it is of my knowledge that the Pinot tool was utilized by dr. Mohamed Soliman, hence it can be easier to troubleshoot some of the problems encountered throughout the project.
This report is divided into several sections, as follows: In Section 2 I will explain the process that I will be taking throughout this internship, explaining the steps necessary for any conclusions to be drawn. In Section 3 I will be describing the utilized software and hardware necessary for utilizing the tools so that it is possible to replicate the experiments I will be performing. In Section 4 an analysis on the projects is made, in order to refine this list into a small number of projects which will be the object of the analysis.

[1] https://web.cs.ucdavis.edu/ shini/research/pinot/
[2] https://github.com/ArchieProject/Archie-Smart-IDE

2 Methodology

As explained in the previous Section, multiple directions could be followed, but for the sake of this internship, and given its short time-span, I will be restricting myself to the second approach.

Firstly, I have read literature on similar projects, and literature that presents the tools which will be utilized in this project. Namely Mirakhorli et al. in [1], [2] and Shahbazian et al. in [3]. In my research project, this literature is important to understand how the design and architectural patterns were extracted, namely in the first two, where the Archie tool was utilized.

Secondly the working environment needed to be according to the needs of the tools. Utilizing the literature mentioned above, and a research paper created by a German research group from the University of Hamburg, named "Recherche zu Architekturanalyse-Tools" [4] I was able to extract some software and hardware requirements for Archie and Pinot to be ran.

Using these two tools, I will be scanning the source code of selected projects, through each of its releases using GitHub's commit history. For each of the commits, an output will be created, highlighting files which contain evidence of GoF [5] design patterns and architectural patterns.

3 Software and Hardware

This Section intends to gives the reader an overview of the attempts taken by me in order to setup a working environment which allowed for the usage of Pinot and Archie. And in the end provide a list of the requirements that allow for a successful usage of these tools.

For the Pinot tool, the author of [4] mentions that the following software/hardware was utilized:

1. Kubuntu 14.04 amd64
2. GNU Make 3.81
3. OpenJDK 6, 7 and Oracle Java SE 8

1. Archie
    - First attempt: Running Archie on Windows 10, using Eclipse 2020-03. Failed
      Successfully created the plugin using Archie's source code. The tool would be created as a Plugin for Eclipse, and appear in the UI, but upon pressing the "Scan" button, no output would be created.
    - Second attempt: Running Archie on Windows 10, using Eclipse Luna. Failed
      Same rationale as above.
    - Third attempt: Running Archie on Linux Ubuntu 18.04 LTS, using Eclipse Luna. Failed
      Same rationale as above.
    - Fourth attempt: Running Archie on Linux Ubuntu 14.04 LTS, using Eclipse Luna and Oxygen. Failed
      Same rationale as above.
2. Pinot
    (a) First attempt: Compiling Pinot on Linux Ubuntu 18.04 LTS, using default GNU Make, G++ and GCC. Failed
    Pinot was not able to compile using the instructions provided by the developers. [3]

    (b) Second attempt: Compiling Pinot on Linux Suse 15.1, using default GNU Make, G++ and GCC. Failed
    Same rationale as above.

    (c) Third attempt: As mentioned in the requirements from [4], I attempted to compile Pinot using Linux Kubuntu 14.04, using GNU make's version 3.81, which is the same as the one they utilized in 2015 upon performing their research. Failed
    Same rationale as above.

    (d) Fourth attempt: Compiling Pinot on Linux Ubuntu 14.04 LTS. Successful
    Using the same GNU Make version as in the other attempts (3.81), and the default (latest) versions of G++ and GCC, Pinot was able to compile and be used for analysis.

[3] https://web.cs.ucdavis.edu/~shini/research/pinot/PINOT_INSTALL

3.1 Final List of Requirements

- **Software**
  - VirtualBox Version 6.0.20 r137117 (Qt5.6.2) hosted on Windows 10.
  - Linux Distro: Ubuntu 14.04.6 LTS, amd64 architecture. [4]
  - Java: OpenJDK 8
  - gcc version 4.8.4
  - g++ version 4.8.4
  - GNU Make 3.81
- **Hardware**
  - 4GB RAM
  - 2 CPU's
- **Utilized Tools**
  - Pinot [5]: It is a tool to detect the GoF design patterns from Java code. The tool is a command line tool, which produces a text output.
  - Archie [6]: It is a plugin to detect architectural tactics. Mostly availability tactics. It is an Eclipse plugin, which supports capturing tactics in Java code.

[4] http://releases.ubuntu.com/trusty/
[5] https://web.cs.ucdavis.edu/ shini/research/pinot/
[6] https://github.com/ArchieProject/Archie-Smart-IDE

4 Selected Project

With this experiment I intend to analyze different open-source projects, and scan them using the tools presented in the previous section, so a condition for these is that they are hosted in a GitHub repository, for ease-of-use.
The list of projects that are present below are the result of a students project, which was created under the supervision of dr. Mohamed Soliman, hence the reason I obtained this refined list of projects.
A part of the extense list was sent to me for analysis, as follows: Apache's - Derby, Cassandra, Hadoop Yarn, Hadoop HDFS, Hadoop Common, Hadoop Map / Reduce, Zookeeper, ManifoldCF, Bigtop, OfBiz, Directory studio, Mina, Camel, Axis2.
A critical factor which is common for all the projects mentioned in this list is the fact that they utilize Issue Tracking Systems e.g. JIRA, to keep track of the issues and, whenever a new commit is made, it can be traced to a specific issue, allowing me to backtrack the rationale behind the changes in the architectural patterns of the system.
These are all projects which are developed under the Apache Foundation, hence they all follow the guidelines necessary to backtrack the changes in the source code, and link them to issues reported over at JIRA.

5 Developed Script

In order to run Pinot, the CLI requests the user to input:

```
pinot file.java OR pinot @listOfJavaFiles.list
```

Taking use of this, I reutilized the script present in

6 Archie Results

7 Pinot Results

8 Discussion

9 Description of analyzed architectural patterns

Appendices

A Outputs from the used tools

10 References

[1] M. Mirakhorli, A. Fakhry, A. Grechko, M. Wieloch, and J. Cleland-Huang, "Archie: a tool for detecting, monitoring, and preserving architecturally significant code," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2014, pp. 739–742.

[2] M. Mirakhorli and J. Cleland-Huang, "Detecting, tracing, and monitoring architectural tactics in code," *IEEE Transactions on Software Engineering*, vol. 42, no. 3, pp. 205–220, 2015.

[3] A. Shahbazian, Y. K. Lee, D. Le, Y. Brun, and N. Medvidovic, "Recovering architectural design decisions," in *2018 IEEE International Conference on Software Architecture (ICSA)*. IEEE, 2018, pp. 95–9509.

[4] D. M. Schablack, "Recherche zu architekturanalyse-tools," Universität Hamburg, 2015.

[5] E. Gamma, *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995.