ARGO Testing Scanner

Architecture Document

Team 5

Developers

Amanda Bosson

Matthew Freese

Jessica Jennings

Mazen Lawand

Hisham Nabi

Abstract

This document will discuss the planned architecture and why it fits our requirements. The main goal of the architecture is to meet expectations for the functional requirements, usability, and constraints of the project. The document will be discussing the style used, model used, and the technology, hardware and software used to accomplish the goals for the product. The rationale for the decision of the chosen architecture is stated in order to reach requirement goals. Architecture has also been chosen in order to comply with configuration management plans for iterative changes to the system.

Table of Contents

[**1. Introduction**](#_4hthb12hkokb) **3**

[1.1 Introduction to Architecture Document](#_skxmx8bp6p) 3

[1.2 Purpose of the Architecture Document](#_6g4f7lcc1aet) 3

[**2. Architectural Style(s) Used**](#_939339bolbtn) **3**

[2.1 Client-Server](#_f5t38d72way4) 3

[2.2 Model-View-Controller](#_6hski9p6g4mp) 3

[2.3 Event-Driven](#_jptusb57pnn7) 4

[2.4 Architecture Support](#_u8n5trpv10d6) 4

[**3. Architectural Model**](#_by0gbipx5iob) **5**

[3.1 UML Component Diagram](#_ti1bd0819uwf) 5

[3.2 UML Activity Diagram](#_7qump455f3kw) 6

[**4. Technology, Software, and Hardware Used**](#_70qmkhiu7dvr) **7**

[4.1 Technology Used to Implement the Project](#_90trmk7ygp47) 7

[4.2 Software and Hardware Required to Support the Technology](#_vqiadaw4s6iu) 7

[4.3 Communication Between Application Server and Database Server](#_88h10cqqquwz) 7

[**5. Rationale for the Architectural Style and Model**](#_3gsohmykxfyi) **7**

[5.1 Rationale for Client-Server Architectural Style](#_n1st1uybrjpq) 7

[5.2 Rationale for Model-View-Controller Architectural Style](#_9dxwozycjxy0) 8

[5.3 Rationale for Event-Driven](#_daapaex1qp7h) 8

[**6. Evidence the Document Has Been Placed Under Configuration Management**](#_tmfv3v9ikfdl) **8**

[**7. References**](#_wrdfvwhoo963) **9**

List of Figures

**Figure 3.1 UML Component Diagram 5**

**Figure 3.2 UML Activity Diagram 6**

# 1. Introduction

## **1.1 Introduction to Architecture Document**

This document details the high-level architectural decisions made for the ARGO Testing Scanner project. It will provide a list of all architectural styles, models, hardware, and software used in order to move onto a more detailed project design and implementation.

## **1.2 Purpose of the Architecture Document**

The purpose of the architecture document is to state the reasoning behind the team’s architectural decisions in relation to the requirements of the ARGO Testing Scanner project. The document will explain how architectural styles and models were chosen, provide basic diagrams of the architecture, and detail the hardware and software needed to meet all requirements.

# 2. Architectural Style(s) Used

## 2.1 Client-Serve**r**

The purpose of the client server architectural style is a structure that is responsible for partitioning tasks and workloads between the servers of some kind of service. Our goal is to have a process running on the server side, performing majority of the processing. The client side of the architecture style will perform the following:

1. Notify the server of user login
2. Provide input data to the server -- handle the user testing the certain application
3. Receive data from the server
4. Notify the server of user logout

## 

## 2.2 Model-View-Controller

The Model-View-Controller(MVC) architectural style is used to explain and explore applications with GUIs. MVC is one of the most common types of architectural styles used for GUI application[2].

The MVC can be divided into three major components:

1. **Model** - This provides the means by which data is retrieved and manipulated.
2. **View** - This represents the visual interface components of the Graphical User Interface (GUI) application which interact with the user. The interaction is of an event-driven nature where actions are initiated via keyboard and mouse.
3. **Controller** - This joins the Model with the View and is the heart of the control logic by associating user-generated events with data actions.

## 2.3 Event-Driven

The event-driven architecture delegates tasks to modules to perform certain tasks. The modules then report back to a central component when a specific event is triggered. Our application will inherit this architectural style by assigning each module a log to scan, and to trigger an event once a specific event occurs.

## 2.4 Architecture Support

The architectures chosen for the application each serve a different purpose - to support various features of the application. The client-server architecture fulfills the requirement of processing horsepower and centralizing tasks in one area. The Model View Controller architecture fulfills the requirement of separation of tasks, that is having the view talk to a controller that handles the request to pass on to the model. The model is responsible for performing the tasks specified. The event-driven architecture helps the application scan different logs simultaneously to detect events specified by the user.

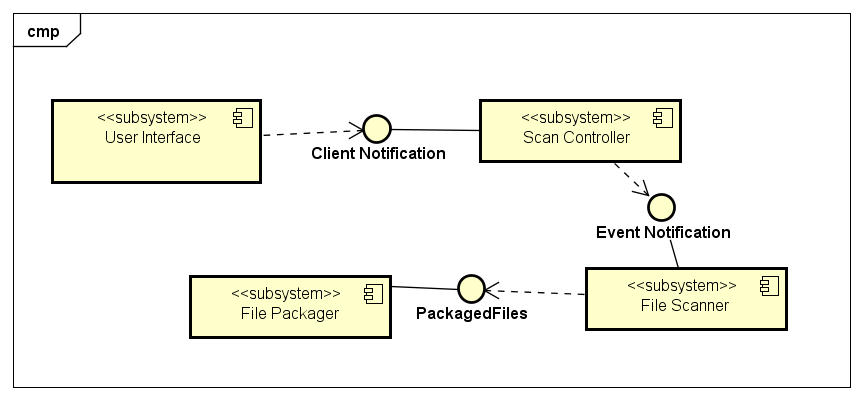
# 

# 

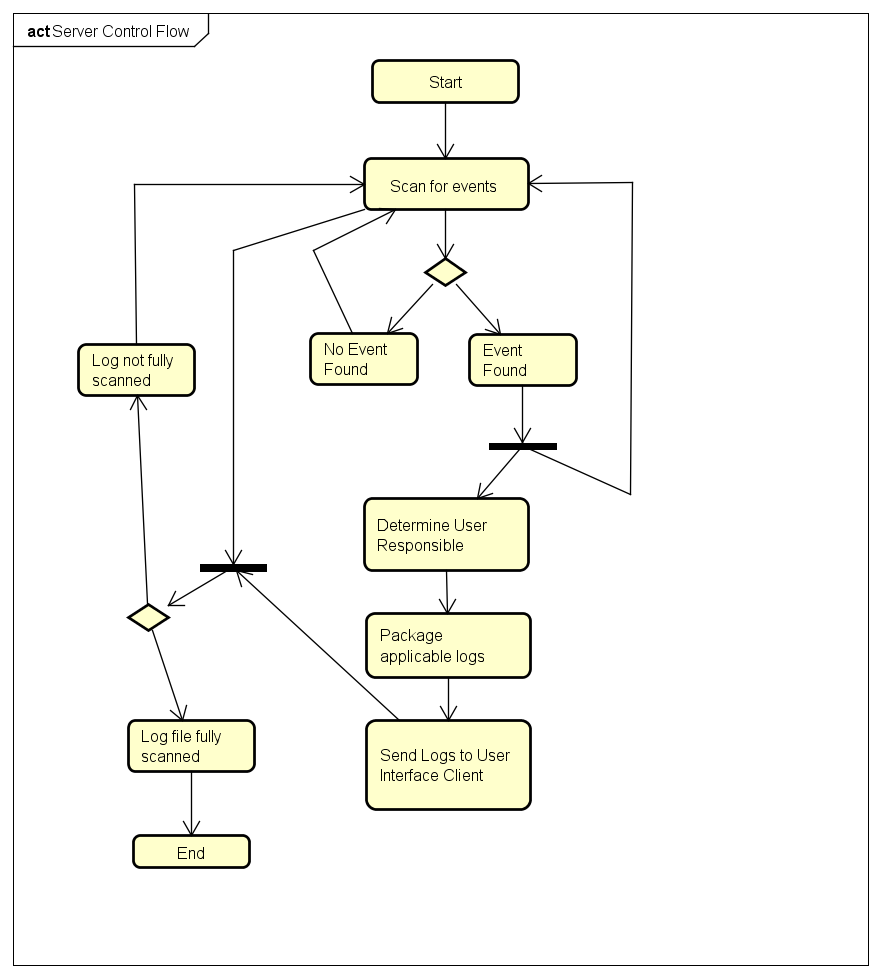
# 

# 3. Architectural Model

## 3.1 UML Component Diagram



## 3.2 UML Activity Diagram



# 

# 

# 4. Technology, Software, and Hardware Used

## 4.1 Technology Used to Implement the Project

1. Java programming language
   1. Works on multiple operating systems
   2. Simple UI creation with JavaFX
   3. Easy to parse XML data
2. GitHub for version control
3. JavaFX Scene Builder for UI creation
4. Eclipse IDE

## 4.2 Software and Hardware Required to Support the Technology

1. Software
   1. Java version 8 update bundled in with application
   2. FindStr application
   3. Hex to ASCII translator
2. Hardware
   1. A computer or virtual machine running Windows 7 OS or later
      1. Must support multithreading

## 4.3 Communication Between Application Server and Database Server

The ARGO Testing Scanner will be a standalone application without a database server, so there is no communication between an application server and a database server.

# 5. Rationale for the Architectural Style and Model

## 5.1 Rationale for Client-Server Architectural Style

The Client-Server architectural style allows for greater versatility and modularity. Using this architectural styles allows us to better divide up tasks between the two components, client and server. It also helps the team to work on each part separately and without interference from one another.

## 5.2 Rationale for Model-View-Controller Architectural Style

The MVC Architecture style is a common style used for applications with GUIs. It allows for greater separation between the design and core logic of the application which increases modularity. MVC also allows for greater flexibility and helps reduce code complexity.

## 5.3 Rationale for Event-Driven

The Event-Driven architectural style is commonly used for systems with asynchronous data flow. This is also best used for individual data blocks interacting with only a few modules. The architecture contains modules that are specialized to deal with specific events.

# 6. Evidence the Document Has Been Placed Under Configuration Management

Configuration management for this document will be handled using github. Changes will be tracked through making commits to this document to keep a record of change dates and the user who made changes.

The github repository can be accessed here: https://github.com/mjfreese/ARGOTestingScanner

# 

# 

# 7. References

[1] <https://techbeacon.com/top-5-software-architecture-patterns-how-make-right-choice>

[2] <http://www.cs.utsa.edu/~cs3443/mvc-example.html>