Architectural Patterns and Styles - Notes

Priscilla Madigoe

08 March 2016

1 Architectural Patterns

1.1 Introduction

Often times source code needs to be organised so that clear roles, responsibilities and relationships of different modules of the code can be well defined. Architectural patterns help in making this possible and numerous types will be discussed in this section. Using these patterns, code becomes easier to maintain, manage and visualise. They also make understanding of how each component works in a system easier. They are reusable solutions to a commonly occuring problem in Software Architecture within a given context.

1.2 Types of Architectural Patterns

- Layered Pattern
- Client-server
- Representational State Transfer (REST)
- Master-slave
- Pipe-filter
- Broker Pattern
- Model View Controller
- Blackboard Pattern

1.3 Architectural Patterns Proposed for PAPERS

1.3.1 Model View Controller

- Background

  In the Model-View-Controller Pattern, an interactive application is divided into three parts: the Model is an object representing the data and activities, the View displays information to the user and the Controller offers a way to change the state of the Model.

- Motivation

  Any form of application that uses a request-and-response logic needs to be decoupled. Separation of responsibilities is useful because functions can work normally when certain subsystems are changed. The Model View Controller makes this a possibility and hence it is chosen for use in this context.

- Componemnts of the Model View Controller Pattern

  1. Model
     The Model consists of the Databse that will be used to store all the relevant information used by the system. For this particular system, Django will be used to handle the Databse. For the Android Application, an appropriate API that will implement REST or Representational State Transfer will be used to communicate with the main Model (Database).

  2. View
     (a) Web Application
     For the Web Application, the browser will be the View because it will display the HTML (with or without CSS and JQuery) that the user will be able to use to interact with the application logic. The Controller will send data to the viewer so that the user can see it.

     (b) Android Application
     For the Android Application, the Android Graphic User Interface, or GUI will be the View and appropriate APIs, such as REST, will be used that will enable this applicaton to manipulate the Model via the Controller.

3. Controller
   The Django Framework itself will be the Controller as it will provide appropriate functionality for data manipulation.