ECE 458: Engineering Software For Maintainability
Senior Design Course
Spring 2015

Evolution 1 Analysis

Brian Bolze, Jeff Day, Henrique Rusca, Wes Koorbusch

Contents

1 Introduction

Good software design is often seen not only as a science but as an art. It is a craft and, like any other type of engineering, is only mastered over time. The fundamentals of good software design, however, remain sound. In this senior design course, we plan on synthesizing our four years of knowledge through the development of a robust and long-lasting software application implementing a web-based calendar. We plan on applying the core principles of good design to our code and to our design process, while continuously evaluating, refining, and improving on our skills of the software engineering trade.

For our project, we built an application using the Ruby on Rails framework. This framework aided us in developing highly modular and reusable code due to the MVC architecture and Rails' powerful web application stack. With the additional help of great documentation and a strong community, we were able to develop a functional product within weeks, despite minimal domain expertise. While our current application has noticeable areas for improvement, our robust model, well thought out API, and extensive front-end templates promise a maintainable foundation for future development.

2 Project Plan

2.1 Design Goals

In order to set ourselves up for a successful project, it was really important to us that we lay out our design goals and priorities before moving forward. We aimed at grounding our design discussions in the fundamentals, which we hoped would help us set up a solid foundation for a project of such a significant size. The core design principles that we focused on primarily were modularity (Open-Closed Principle), re-usability, and the DRY principle. Another consideration that we put a lot of thought into was our language and framework choice, which will be expanded on in the next section. Besides promoting good code design, a well-chosen framework would allow us to get up to speed quickly, which was another important criteria for our team given our experience level with web applications. During our initial stages, a particular focus was placed on developing a robust, accurate, and extensible model for our data. This would lay the groundwork for our API with the front-end components, which we saw as vital to our success down the line.

2.2 Language Choice

The language decision was extremely important for our group for a few reasons. Firstly, no one in our group had previous experience in developing web applications. Accordingly, we stressed finding a language and framework with a strong community, good documentation, and abundant resources. Secondly, seeing as our project would be maintained over an entire semester, we strived to find an environment that would support and promote good design practices. To do this, we would need a language with strong object orientation and a framework with a highly modular architecture. Lastly, we also needed the flexibility to implement our application with high customizability to meet any obscure requirements. After extensive research and testing, we found that Ruby on Rails was able to meet our needs better than any other framework.

2.3 High-Level API

stuf stuf stuf

3 Design Review

3.1 Status

stuf stuf stuf

3.2 Design

stuf stuf stuf

3.3 Alternate Designs

stuf stuf stuf

4 Next Steps

stuf stuf stuf