ECE 458: Engineering Software For Maintainability

Senior Design Course

Spring 2015

Evolution 2 Analysis

Brian Bolze, Jeff Day, Henrique Rusca, Wes Koorbusch

Contents

1 Previous Design Analysis

Considering our groups general inexperience with Rails, we as a group were generally very happy with how our previous design choices impacted our ability to fulfill Evolution 2 requirements. One key design feature that we created in our model that made some things very simple for us in this evolution was the idea of a subscription. At the time, we believed that having a user subscribe to a specific event would make notifications, requests, and updates easier to add in the future. This particularly helped with things like email reminders. A second design decision that we unfortunately unable to implement in Evolution 1 but had always planned to include was a visibility model object. There was a significant discussion over whether or not visibility should simply be an aspect of the event field or whether it should have its own entire definition. The fact that we planned to have it as a distinct definition in our design made it easier to incorporate the ability to see events or not depending on user choices to requests and invites.

One specific area where our design was poor was with repeating events. We thought the implementation would be relatively simple: make copies of a single event at some defined period. However, there are a lot of adjustments that need to be made when editing a repeating event. Initially, we attempted to make a simple encoding of a single number to the days of a week that an event could be repeated and also the period that it would be repeated. We then realized, it was very difficult to do math with regards to a calendar with integers that represent days of the week. As is clear in our Evolution 2 implementation, we refactored our design with regards to repeating events in order to make it more robust and flexible.

2 Current Design Evaluation

3 Future Design Needs

4 Design Process Notes

This section of the document describes experiences regarding the design and testing process for each member of the team. These experiences include analyzing data, design processes, and team management.

4.1 Designed and Conducted Experiment

Jeff's Contribution

One key front end aspect that we wanted to add to our web application in Evolution 2 was popup forms and views. As a team, we agreed that these types of views would not only would to a more fluid user experience, but generally look more modern. In order to implement these popup forms (along with other various UI features), we added a Rails Gemfile called Foundation. This Gemfile allows for easy inclusion of many jQuery elements in the views files of our Rails app. Using an aspect of Foundation called reveal-modal, I attempted to implement the form to create a new group into a popup window that would allow the user to create a new group without

leaving the index page. I created the form and could submit without error, however the users were not being added to the group in the actual database. After, attempting to modifying aspects of the reveal-modal, I instead tried implementing my new group form in a simple rails view page instead of attempting to add dynamic popup views. When trying a normal form and realizing that the database was successfully populated with that form, it was obvious that the issue was with submitting data using reveal-modal. After more research, it became obvious that to do form submittal in a popup view, we need to utilize AJAX with dynamic page loading.

4.2 Analyzed and Interpreted Data

Jeff's Contribution

Throughout our teams development in the Rails framework, the ability to read and understand messages outputted from the rails server has become incredibly important, specifically with debugging. My previously mentioned bug involved not only designing an experiment, but also interpreting specific terminal data in order to successfully debug. The specific issue was that the checkboxes associated with the form were not being properly read by Rails as a result of being in the reveal-modal window. With functioning checkboxes, the successful creation of a group results in the terminal outputting a visualization of a map associating the id of each user with a one or zero (representing whether they were added to the group or not). Originally I did not realize that this was the desired terminal output; I only new that the terminal was outputting that the addition of users to the groups members column was forbidden. However, after reviewing the output of a similar form that Henrique had implemented, I was able to see what the correct output should look like and deduce that the issue was with Foundation, not my implementation of the form. Only through the ability to understand and analyze this sort of server data was I able to completely debug this issue.

4.3 Designed System Component to Meet Desired Needs

Jeff's Contribution

4.4 Deal with Realistic Constraints

Jeff's Contribution

As always with large scale CS projects as an undergraduate, my team was constrained by time towards the end of evolution two work period, especially as a result of other midterms and big assignments due among the team. As a result of this fact, we had to ignore some smaller bugs in order to work on other more significant ones. For example, one of my jobs towards the end of the evolution period was to discover why our calendar would only render after going to the main events page and then refreshing. However, at that point in time, we still did not have recurring events working completely. At that point, Henrique and I decided that it was more important for me to assist him with developing methods to make recurring event creation easier than it was to make the calendar render on every event page click.

4.5 Contributed to Team Work and Interacted with Team Members

Jeff's Contribution

There were two primary tracks for interacting with my team members: through project management tools and in person. Using tools like Trello, Facebook Chat, and useful commit comments, I did my best to always keep the rest of the team updated on how progress on my responsibilities was going. This not only allowed team members to assist me if they saw any flaws in the work I was doing, but also just kept the entire team on the same page. Throughout this evolution, the primary way that my team and I worked well as a team in person was through pair programming and peer code reviews. Often if one of us had an issue with some aspect of the project, a fresh pair of eyes or a new way of thinking about some specific algorithm quickly solved the problem. This sort of team interaction not only allowed me to help my teammates, but also allowed the to help me in a very effective manner.