ECE 458: Engineering Software For Maintainability

Senior Design Course

Spring 2015

Evolution 3 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

On Evolution 3, we were able to reuse several models to tie up the new logic while creating new structures and engendering more code. The team has improved on AJAX and jQuery skills to add more dynamism to views while the backend still tries to maintain its cohesion.

2.1 views

Views have improved significantly since evolution 2. We were able to group several features into forms in the home screen, but time constraints did not allow us to apply the technique to all functionalities. It is possible we migrate everything to a more user-friendly interface by the end of evolution 4. For the two new functionalities, time slots and free time, partials are still a crucial feature. We still reuse similar html code and modularize it among partials.

2.2 Controllers

The `TimeSlot` controller is relatively short since the validation, creation, and deletion logic is moved to the models as always. The controller basically handles creation, deletion and slot signup. It does not even handle selection of free slot times, that logic is moved to the models as well

The `FreeTime` controller, however, handles heavy logic and computation. Since a FreeTime