Requirements and Schedule Postmortem

Features & Cuts

We were fortunate enough to be able to make no core feature cuts, and even include some stretch features we were not sure would be possible in the time allotted. We had promised the following features by the 1.0 release in our schedule:

1. Pin dropping/reading on a Google map
2. Pins outside the specified unlock radius are unreadable
3. Integration with Facebook
4. A settings menu and help screen
5. A profile page that tracks posts made and viewed

We were able to provide all these features and more, namely:

1. Pin filtering based on different categories (viewed, locked, posted by friends, etc.)
2. Picture posts
3. A “wait” signal displayed when a post is in the process of being posted to the DB

Stretch features we had discussed but were unable to provide:

1. Pins with video messages
2. Deleting pins automatically (Pins time out) and manually
3. Logging out
4. Ranking system for pins
5. Scavenger hunt (linking pins together)
6. Post to your Facebook wall
7. Limit audience of posts (Public, Facebook Friends, specific Facebook list, etc.)

We believe that our initial schedule may have been conservative in which features it promised by the 1.0 release. However, we believe this approach to scheduling was wise, and has put us in a position where we are delivering more than we promised. It would have been easy to schedule all the stretch features we could imagine, but then we may have ended up short of our commitments.

Task Assignments

Andrew: Originally, I had planned to work on the database code and unit tests. I ended up spending a lot of my time on these areas of the codebase, though I spent less time on the unit tests than I had expected. Because of the difficulties we had setting up Parse, the amount of work I spent trying to figure out technical issues was much greater than I had anticipated. I also worked a great deal on reviewing comments and code, both in the database code and in other modules, to ensure consistency across our code. Overall, I think I did a good job of balancing the amount of time I spent in each of my roles, though I think I should have spent more time working on unit tests earlier on in the development process, as I feel these could have been better done.

Duncan: Initially, I volunteered for the task of managing test frameworks and larger-scale system testing. As the quarter progressed, I began to regret that decision. As it turns out, going into system testing solo without requesting additional help from my team members led to a final product that was below standard. In retrospect, I should have taken steps to ensure that I understood the task at hand before assuring others that it was on track. I ended up getting some help closer to the end, but by that time came, I was already swamped with my other classes, leading to improved, but ultimately sub-par contributions on my part.

Ethan: My role was originally to work on Google Maps and the Android user interface associated with the map. This turned out to be very similar to what I actually worked on. I mostly focused on handling user input to the app and responding appropriately. What was not expected was the amount of time spent fixing bugs. This entailed working on other areas as well to fix bugs that needed to be fixed before a deadline. Looking back, I believe I should have spent more time on unit tests during coding as opposed to saving the majority of them for later.

Katie: I had originally planned to be working on the frontend with maps and the UI. While I did do a few things including making a logo and some backgrounds, I didn't do much frontend work. I ended primarily working on the model classes along with the unit tests for them. I also tried to keep our UML diagrams up to date over the course of the project. I spent too much time working with Eclipse errors during the beginning of the project. I never ended up figuring out what was causing errors but after installing a new version of Eclipse my problems were solved, so perhaps it was just too old. Writing test classes also took much longer than expected because Parse doesn't have very good support for testing. I wish I had spent more time figuring out how to test with Parse early on, or finding workarounds to do so that the tests for these classes would be more robust.

Matt: Originally, I was assigned to work as a developer on the Google Maps/Location and working with markers on the map. While I still worked on that team for most of the project, my role became more focused on extra features and usability. I was the lead for implementing posting photos, changing to LocationClient service, and adding loading wheel. We probably spent too much time on FB Login because most of the pin filtering was dependent on FB username. I wish I would have spent more time making unit tests; our test infrastructure was ill-defined so it would have really slowed down progress to try and figure out Android testing in addition to development.

Megan: When we first delegated roles, I was assigned to work on databases. Although I did spend a lot of time working with the Parse database and Parse's Android SDK, I probably spent more time working on Facebook integration. Facebook integration took a surprisingly large amount of time. Initially, our team estimated that Facebook integration would take significantly less time to implement compared to other features. Unfortunately, we encountered several major issues while integrating Facebook with our app, so we had to spend a considerable portion of our total development time working on this feature. Looking back, I don't think any of the major

issues we encountered could have been prevented by additional testing. Most of these problems were simply due to a lack of experience integrating Facebook with a mobile application.

Mike: My initial role in the team seemed to be as a developer working on Android activity design and local user post storage. I worked on setting up the initial code suite and putting together the functional activities that comprised the basic app. I think that most of my time was occupied by gaining understanding of android architecture, and creating logic for translating post objects to the users view. This remained true throughout the project though I did bridge from this role to fulfill other tasks. If there is something I did not spend enough time on, it would probably developer written unit tests. Unit tests were difficult for the whole team and I think I could have taken more time to write basic tests for the code I wrote. I also wish I had taken more time in the very beginning of coding to make sure everyone was using consistent coding practice. This lead to inconsistent code style and branch use that we had to clean up. I wish I had spent less time on setting up automated test builds via Jenkins. This turned out to be incredibly hard to setup and was something I never got working due to complications with android emulators in Jenkins.

Neil: My initial role was PM and general developer. We did not specify development teams until midway through the quarter. Once we did, I volunteered to be on Test. I spent the majority of my time doing PM-type things, writing documentation, and generally trying to make sure information was consistently flowing across the group. I also spent a significant amount of time on a single bug. The Android emulator bug took approximately 15 hours alone to resolve. I spent a good deal of time, more than I expected, working on tests and trying to get test frameworks like a coverage tool set up. I wrote less code (and almost no non-test code) than most members of the group, and I would have liked the opportunity to write more.