

Post Mortem Meeting Notes **Shumoku 種目**

Hitarth Patel, Richard Lee, Kristen Kho

June 9, 2008
CSE 210

Setting

The Shumoku post-mortem review meeting began 1PM PDT on a beautiful day in the Carmel Valley/Del Mar area of San Diego, California. The meeting was catered by Pat & Oscar's with pasta, salad and breadsticks.

Technology

In our early iterations, we ran into many technology issues. This was one of the big reasons for the project failing to develop more advanced features. We never completely set up a working local environment on each of our computers. Since PHP is configurable, we often had issues with our local machine's configuration being different from our hosting server's. The free PHP IDE, Eclipse PDT, had very bare functionality. Features such as refactoring and automated upload would have saved development time. A commercial IDE like Zend Studio may be considered for future development. (PDT is also developed by Zend, so we believe PDT's lack of functionality was intentionally done to lead us to buy Zend Studio).

We learned that the learning curve for some chosen technology was quite steep. Although PHP gave line numbers to where code would break, the messages were often very unclear. Nuances to the language like dynamic typing, variable scope and the way PHP handled includes proved to be counter-intuitive. JavaScript, as pointed out by Bill in our technology plan comments, was also difficult to debug. Time spent debugging and looking up reference material took away from time spent on development. Interfacing with third-party technology was also a big challenge, causing delay in our more advanced requirements; text messaging, one-click adding to personal calendars and Facebook applications for example, were deferred to later releases.

Some technologies yielded positive outcomes, which we plan to keep for future use. Google Maps was reliable and its website contained many good examples. Although its reference material was incomplete, Google provided forums for more advanced and undocumented techniques. SVN setup was delayed to administrative issues with UCSD's CSE Help. However, once setup was completed, development greatly sped up. Division of labor became much easier and communication could be done through logs instead of more costly face-to-face meetings. Google Docs reaped large benefits as a collaborative tool for writing milestone design documents. Rather than all of us talking and having one scribe, we each brainstormed aloud as we entered our ideas into the document. Each member of the design team could also add more as the week went on. Google Groups was used to privately store our shared information, while PBWiki shared our public information. Starfield's MySQL client, supplied by Kristen's GoDaddy account, allowed us to manually enter data and alter our tables. This saved in the overhead of looking up the full MySQL statements.

Scheduling

Our team consisted of three people developing part-time. Each member had classes and work competing for their attention. The overhead to switch among these responsibilities was a factor in slowing down development. We also had to meet at night when our minds were drained by all our other work during the day. Ideally, Shumoku would benefit from having enough funds to have full-time engineers.

The team met two times a week, usually Tuesday and Thursday night. At meetings, milestone documents were written collaboratively and division of labor was discussed for each member to explore individually. Since some technology was new to the team, strict deadlines could not be made or be depended on. Although we had ambitious goals early on, we had to reprioritize our requirements to more modest goals.

Team Shumoku did work well together. As a small group, the communication overhead was very small. Despite our busy schedules, each member was very willing to meet often and stay late. Positive attitudes and good team chemistry helped us push through all our scheduling obstacles to finishing all basic features.

Process

In our early planning process, we decided to use a spiral process that integrated some features of agile development. Our goal was to have a process that iteratively developed our software and helped us evaluate risks. We found out that the process would also be developed iteratively.

The first iteration consisted of developing a prototype. This helped us realize some of the technology issues, like how to use Google Maps and integrate the MySQL database. It provided for a nice basic proof-of-concept. It also let us get feedback from users. This prototype became a throw-away since it was developed with no real architecture in mind; database queries were hardcoded into the PHP and searching was omitted.

The second iteration was designed using the three-tier MVC architecture, with each team member leading a layer. Each lead was responsible for delivering their layer and scheduling integration meetings with other leads. This worked well for division of labor and testing. The UI side has already reaped the benefit of adding a second UI component for mobile devices. The integration meetings consisted pair/team-programming, code review and the addition of more test cases. These meetings helped to discover simple semantic and logical defects.

Risks

1. Adding New Team Members/Attrition
2. Adding New Features/continue development
3. Technology Burdens
4. Getting User Feedback
5. Establishing User Base
6. Future Funding

Risk Resolutions

1. Two new communication initiatives:
 - Increase use of pair-programming for current leads to ease new members into the current code.
 - Commitment to documenting design and code so that new team members can learn quickly and knowledge is not lost from departing members.
2. Explore features for feasibility and time estimates. We already have a list of customer requirements. We will continue to add and prioritize them based on user feedback.
3. Research into alternative technologies required. Funding for commercial software tools to be considered.
4. We are currently doing user surveys to help prioritize additional features. Other methods of communication will be considered to add depth to analyzing the user's needs.

5. Explore marketing opportunities like speaking with student organizational leaders, flyer and chalking.
6. Look for venture capital funding or revenue streams.

Risk Results

1. The following standards will be added to our process:
 - Commenting and design documentation. Eclipse already automates comments for each function, so developers will just need to fill in the blanks.
 - Code review system.
 - Instructions to set up common development environment including IDE, Apache HTTP'D Server, PHP (with proper php.ini) and MySQL Server.
2. Continued enhancement of our iterative process:
 - Stricter deadlines and standards for proof-of-concept on new features. Our risk-driven approach to design will aid us in prioritizing which features are to be added.
 - Test-driven approach.
3. Future tools and technology meeting has been scheduled to access current technology. We plan to request funds from SWVC dedicated to software that speeds up our development.
4. Future funds will be dedicated for marketability surveys. This will provide much more depth in user feedback than our current survey feedback system gives us. Our team also plans to design a user feedback agent component for collecting feedback directly from the user. Future expansion can include establishing specialized teams like product management and business systems analysts to get customer requirements and use case scenarios.
5. Develop a marketing plan with funds being dedicated to marketing analysts and publicity.
6. Future funding sources:
 - SWVC
 - Advertising
 - Corporate fees for pop-out effects

Commitment

- Detailed requisition reports to SWVC for future funds dedicated to full-time hires, commercial tools and marketing.
 - Demonstration to SWVC on June 9th, 2008.
- Plan schedule for new release scheduled for September 2008.

Conclusion

Although we are proud of Shumoku and what we've learned in the process, we learned that much needs to be improved. As an educational experience, Project Shumoku was a success. We understand, however, that our accomplishments were modest and can strive for more. We need to continue to evaluate technology through the lifetime of the project to improve all phases of development from design to testing. From our experience with Shumoku, we learned to better estimate our next schedule release. Most importantly, we will continue to evolve our process to better evaluate our risks and listen to what our customers want.