Reflection - Software Engineering

Scrum

The development process followed the agile method known as Scrum. The list of features that were to be implemented was decided at the beginning of the process, after which the core of the application was designed together. When the core was sufficiently discussed, *sprints* started, each one decided to last one week. At the beginning of each of these sprints, features were chosen from the list and responsibilities were set for each member. At the end of each sprint, the sprint was discussed and planning for the next one was started. Working in sprints became routine, and after each sprint the team members became more accustomed to the work flow of taking something incomplete and trying to implement it with a deadline, as opposed to working with lots of different tasks simultaneously until the end of the project.

Meetings

Meetings, called *daily scrum*, were held during this process multiple times a week in addition to the sprint planning and retrospective meetings. These were held mostly physically, but also through Skype whenever appropriate. They led to a better understanding of the system as a whole, when each member was usually working individually with a set number of tasks, which led to better code review sessions as the project went on.

Goals

The goal of the sprints were to motivate by the sense of responsibility, and their purpose was to create a set of complete features each week to integrate to the existing system. In our case, productivity was increased because of this sense of accountability. It worked well at the beginning, but proved difficulty as the project grew, when new systems were more difficult to integrate. Eventually, time was spent mostly ensuring that the new systems worked with others, and that bugs were fixed, which caused a lot of stress making other tasks, such as writing tests and adding more features, fall behind. However, the project was planned with scalability in mind, which made the cancellation of later milestones possible.

Workload

Since the workload was divided, the team had to work separately to ensure efficiency. This proved to be impossible during the later phases of creating the application core, as it required a lot of work, and would have been time consuming to get into. Consequently, a development branch was created, where the team could add any updates relevant to their assignment, and only features which were strictly independent were implemented separately in different branches. To ensure minimal issues, such as two members completing the same task, these assignments were thoroughly discussed several times a week. Creating fixes to the core of the application also meant that dependent features (branches) needed to be merged to the development branch. This made working as a team more difficult, however, these necessary redesigns led to thoroughly examining the core and rewriting it. This, of course, resulted in issues. To fix them, the exhaustingly defensive method of throwing exceptions and logging every event was used, which made implementing new features much easier again as the core was improved upon.

Conclusion

In conclusion, if the project were to be restarted, more time would have been spent in the later phases of designing the core, and thorough planning and writing of interfaces would have increased the chances of subsystems working together. But since the chosen method of development, Scrum, is iterative, this would have meant less time for sprints, which in turn would have meant fewer sprints and thus fewer new features chosen. Consequently setting separate tasks for each member would have been more difficult.