JAGSAT project

Postmortem analysis

Juan Pedro Bolívar Puente
Aksel Junkkila
Guillem Medina
Sarah Lindstrom
Alberto Villegas Erce
Thomas Forss

Project Course
Åbo Akademy

March 26, 2010

Revision history

Date	Version	Description	Author
21.03.2010	1.0	Creation	Sarah Lindstrom
22.03.2010	1.1	LaTeX Format	Alberto Villegas
25.03.2010	1.2	Tuning.	Juan Pedro Bolivar

Contents

1 What went right?

Considering how big our team is, an issue that usually leads to misunderstandings and problems, we have manage to communicate fluently, discuss correctly and take the proper decisions for a successful project. Second, we were dealing with a project that required from us some knowledge that most of us we did not have. We learned a lot, improving our skills and chances for success. Third, we showed how a mixed group of Finnish and Spanish students can learn, design, program and develop a project together.

With that said, we all agree that we have a great team. We are a mix of different skills and we all have experience of both video and board games, which makes us a good team for this particulate project. We all had fun developing this game and according to us it has gone quite according to the initial project plan. Some small setbacks were expected but overall we think that our project was well executed.

The communication between the project members has been good for the most part and we have had quite frequent meetings. The work task division has also been good, although one of us who knew Python from before have acted as a ”mentor” for the rest of the team and therefor done a bit more of the programming. We have all participated in creating the documentation required for the course.

Also, we are very happy with the design of the basic framework. In a previous status presentation one of the lecturers asked us whether we might be suffering with the over-design problem: we could end up with a nice generic library but no product. Actually, most of the high level logic has been added to the game in the late phases of the development in a very short time thank to this framework. It is still not perfect, and the interface impedance with TF was tougher than we expected, but it has been an invaluable tool in being able to provide a robust, extensible and functional software.

2 What went wrong?

Something that at the beginning was supposed to be fundamental for us, based in what we have learn in many courses along the years, has turned out to be meaningless big documents. I am talking, of course, about the

exhaustive documentation we prepared. Obviously, it is necessary to have good documentation for a big project, but in the end we did not use most of the amount of information specified there. It is also because another of our problems, we tried to do much more than what was possible for the time and expertise our team had. A ironic problem, since the time we spent doing documents is, probably, the time we could have needed to program the now missing modules.

Finally, one unexpected problem (but anyway reflected in the project risks) was the appearance in the market of the iPad device and the repercussion this hardware had for the company that contracts us. Suddenly we were facing the chance of being without any prototypes to use for testing and showing.

2.1 Tower defense part

One thing that did not go exactly as we had planned from the beginning was the implementation of the second part of the game, the tower defense fighting part. This was not implemented mainly due to lack of time. The basic part of the game took a bit longer to finish than we had planned and at about halfway through the project we decided to focus on the risk part and making that work well instead of doing two games that would not work as we would have wanted them to.

In December we evaluated the remaining time table and decided to focus on the Risk part. At that time we were not sure if we would have the time to develop the tower defense part of the game, but just in case we set our deadlines a bit earlier than needed. We did this so that we could be prepared for the alternative of actually completing the tower defense part, if we would happen to get the risk part done in time.

In March we concluded that completing the tower defense part could be possible, but it would be at the cost of important features in the risk part and therefor we confirmed that the focus should stay on that part of the game. We were aware of this risk from the beginning and according to us we spotted the problem in time to adjust the time table.

Note that this was not just bad planning on our side. The Tower Defence part had the additional risk that it depended on a multi-touch screen. In our early discussions with TribeFlame they suggested that their device would have two-point clicks and that, while their library did not support multi point interfaces at the moment, they would aid thas in adding the

necessary support. In the late months of the development when we decided that we had no time, we where highly influenced by the fact that the TribeFlame prototypes did have no multi touch support and that having to add multi touch (a bit experimental functionality on X Server still) support to both the PySFML and TF library on our on would have been impossible.

2.2 Python as programming language

Another thing that didn't go wrong, but took a little more time than we had expected, was the learning of Python as a new programming language. Python was new to many of us and it has required many consultations and teaching sessions for everyone to understand how the language is structured. We all agree that in retrospective it has been a very useful and good way to learn a new programming language, as well as educational.

Also, the library that provided widgets and other user interface facilities, provided by TribeFlame, was unexpectedly buggy and was almost completely undocumented. Because we had the source code, we could fix the bugs and add the missing functionality ourselves and also derive the interface properties, but this has been very time consuming.

3 Lessons learned

The design documents specifies some modules that we where unable to program due to a too fixed schedule. Anyway, we had this problem because we were most of the time dealing with the lack of documentation our third party libraries and API had. This is one of the lessons we learned, when you depend of a company who provides part of the code, to keep a fluent contact must be one of the first objectives.

As mentioned before, probably the most important lessons here are learnt from the problems we had. Communication with the company and the many chances of failure you have when depending of third parties turns to be a bit disturbing. Anyhow, this is something real that we will have to face in the future. Documentation has proved to be important, but more important still is to be able to estimate and prepare a good schedule, without it the documentation could be useless.

Finally, lack of documentation has shown us how important it is for the people that are forced to use what you program to have a clear idea of the purpose behind the program, otherwise they will lose a lot of time figuring out things that probably are simpler than expected.