Project plan + study diary

Yet Another Game (YAG)

version 1.11

TUT	Pervasive Computing	TIE-21106 Software Engineering Methodology
Author: Group 22		Printed: 14.02.2017 20:04
Distribution: Farshad Ahmadi, Tero Ahtee, Kari Systä		
Alexandre Kirszenberg, Milla Mäkinen, Mengyang Chen, Daniel Berezvai		
Document status: draft		Modified: 14.02.2017 20:04

VERSION HISTORY

Version	Date	Authors	Explanation (modifications)
1.0	28.01.2014	Marko L.	Initial version
1.1	11.02.2014	Marko L.	Deleted finnish text
1.2	18.01.2015	Tensu	Sections 1.4.x, cosmetic tuning
1.3	26.1.2015	Marko L.	Final toucher
1.4	16.01.2017	Kari S.	Adaptation for 2017 needs
1.5	25.01.2017	Daniel B.	Initial shared document version bump
1.6	29.01.2017	Alex K.	Expand on risks.
1.7	29.01.2017	Chen	Process description
1.8	29.01.2017	Daniel B.	Final touches, layout fix, merge
1.9	29.01.2017	Milla M.	Tools, technologies, fixes
1.10	14.02.2017	Alex K.	Sprint 1
1.11	14.02.2017	Daniel B.	Doubled the length of each paragraph in our sprint 1 review. Detail rich explanations, miscellaneous fixes, and spellcheck. Clever titles introduced (I remembered to update Table of Contents too!). New tool: LaTeX. (A. K.)

TABLE OF CONTENTS

1. PROJECT RESOURCES

This chapter holds the project resources.

1.1 Personnel

- Milla Mäkinen <milla.makinen@student.tut.fi>
  Scrum Master, Programmer & totally an artist. Industrial engineer, programmer at heart. Some five years of game dev experience mainly in C++ and Javascript. Specializes in having no life.
- Mengyang Chen <mengyang.chen@student.tut.fi>
  Coder familiar with Javascript and C++, interested in web programming and game programming.
- Daniel "3ICE" Berezvai <daniel.berezvai@student.tut.fi>
  Product Owner, Game modder (Warcraft 3, etc.), studied 20+ programming languages, personal website: http://3ice.hu/
- Alexandre Kirszenberg <alexandre.kirszenberg@student.tut.fi>
  Programmer, Previous experience as a Frontend Software Engineer. Interested in Software Development.

1.2 Process description

The goal of our project is to make an interesting game, immerse ourselves in the SCRUM methodology, and of course pass the course and get the credits. To measure the success of the project, we will evaluate whether all the user stories are fulfilled at the end.

Basically we are going to use WhatsApp for discussion with each other and allocate the works evenly to everyone in the team, and use email to inform some general information to everyone in the team.

To ensure the success of the project, one will inform other team members when facing some really tough tasks, so coding nights or jams will be held to solve the task together.

1.3 Tools and technologies

The tools used in this project are listed in table 1. Most of them are not affected by versioning as everyone is automatically updated to the latest version. If version difference issues arise, the contact person will decide which version everyone in the team will use.

Table 1.2: Tools used in the project.

Purpose	Tool	Contact person	Version
Documentation	MS Word (word processing) office.microsoft.com	D.B	2013 -or- 2016
	MS Word Online (shared editing) Office 365	D.B	N/A
Communication	WhatsApp, http://www.whatsapp.com/	M.M	any
	Skype (internet calls) http://www.skype.org	M.C	any
Version management	GitLab http://rd.gitlab.tut.fi Accessible through the course selection or https://gitlab.rd.tut.fi//sweng-2017/g22---yet-another-group	M.M	N/A
Agile Management	AgileFant https://app.agilefant.com	M.M	N/A
Programming language of choice	Processing https://processing.org/	D. B.	3.2.3
Documentation	pdflatex	A. K.	1.4

2. STUDY DIARY

This chapter holds our journal of lessons learned during the course. Detailed analysis of each Sprints' contents shall follow.

2.1 Sprint 1

For this first sprint the product owner proposed a very minimalist approach as we familiarized ourselves with the several new tools at our disposal. The official requirement was to fulfill two user stories per sprint, and conveniently enough, the first two tasks were as simple as; one: reading input from the user (their name) and then two: printing out a personalized backstory for them.

2.1.1 Everything went well (almost everything)

Right from the get go we had a very good understanding of how to achieve our goals. As such, it was easy to start working on the project and rapidly deliver outstanding results for the first sprint. In fact we completed our intentionally very light sprint 1 workload before it even officially began.
As to not fall into despair from lack of work, we selected additional tasks to complete. In the end, we have four additional user stories more or less in a completed state, in addition to the first two proposed solutions that were refactored several times until they met everyone's ever higher standards for good code.

2.1.2 We had difficulties — communication and missed deadline

Working in a team can be difficult, especially when it comes to communicating between team members. None of us were very familiar with Agilefant, so we did most of the communication on WhatsApp and in person.
One failure was that we missed the delivery deadline. All our tasks were complete weeks ago, but we never submitted them to Repolainen because we constantly kept adding new things. In the future, we will create calendar entries for each deadline with automatic email notifications set up to give us a reminder two days in advance.

2.1.3 What were the main learnings

We should use Agilefant more often, in order to make sure that we’re all on the same page. It also helps in defining clear objectives for individuals and for the team as a whole.

2.1.4 For the next sprint, we decided to change our work habits and one tool

We decided to try and take more advantage of the tools at our disposal. Since our team works mostly in quick and very productive iterations, it might be interesting for us to synchronize our efforts in real-life sprints.
As we have had difficulties taming Microsoft Office, a proposed move to TeX was initiated. It will improve our work process greatly. No more crashes due to faulty edit conflict resolution, nor unnecessary struggling with the laggy online version of Word that performs especially poorly on MacBooks.

2.2 Sprint 2

2.2.1 What went well

2.2.2 What difficulties we had

2.2.3 What were the main learnings

2.2.4 What did we decide to change for the next sprint

2.3 Sprint 3

2.4 Sprint 4

3. RISK MANAGEMENT PLAN

Table 3: Project risks.

Risk ID	Description	Probability	Impact
P1	Short term absence	3	1
P2	Long term absence	2	1
P3	Someone drops out	2	2
T1	Someone force-pushes to Gitlab	2	1
T2	Processing turns out to be a very bad choice	1	3
C1	Customer changes requirements	3	2
M1	Divide between management and personnel	1	3

3.1

3.2 Personnel risks

3.2.1 Risk P1: short term absence of one person

Root cause: A member will be absent for several days.
Importance: Little importance, one of us can manage the project on their own anyway.
Avoidance: It would still be nice to warn the project members so that we don't rely on the concerned person to do any work.
Response: Someone else takes responsibility for the person's work.

3.2.2 Risk P2: long term absence of one person

Root cause: A member will be absent for several weeks.
Importance: Similarly to a short term absence, we can manage without one person for a prolonged period of time.
Avoidance: A warning will do.
Response: Someone else takes responsibility for the person's work.

3.2.3 Risk P3: someone drops out

Root cause: A member drops out of the course
Importance: This is slightly more impactful then a prolonged absence.
Response: We will have to reorganize the project around the three or fewer remaining members.

3.3 Technology risks

3.3.1 Risk T1: someone force-pushes to Gitlab

Root cause: Lack of knowledge in the technology brings someone to erase all progress on Gitlab.
Importance: Little importance, other members will have a copy of the project's history.
Response: Find out whoever has the most recent copy and push again.

3.3.2 Risk T2: Processing turns out to be a very bad choice

Root cause: As we iterate over the project, we realise Processing will severely hinder our progress moving forward.
Importance: Extremely unlikely given our constraints.

3.4 Customer risks

3.4.1 Risk C1: customer changes requirements

Root cause: The customer changes their mind on a part of the project.
Importance: Will depend on the size of the change.
Response: Create or modify user stories, rework and refactor the concerned parts of the project.

3.5 Management risks

3.5.1 Risk M1: divide between management and personnel

Root cause: The management and the personnel do not see eye-to-eye.
Importance: Very unlikely considering the management and the personnel are one and the same on this project.