Project plan+study diary

Project name

version 1.5

TUT	Pervasive Computing	TIE-21106 Software Engineering Methodology
Author: Markus Ylisiurunen		Printed: 10.03.2018 12:47
Distribution: Markus Ylisiurunen Tuomas Pekkanen Jere Metsäranta Pedram Ghazi		
Document status: draft		Modified: 10.03.2018 12:47

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	08.01.2018	Farshad A.	Adaptation for 2018 needs

TABLE OF CONTENTS

1. PROJECT RESOURCES

This chapter holds the project resources.

1.1 Personnel

Tuomas Pekkanen, Team Member
Has prior experience in Unity, C# and C++ in small scale projects and from a summer job. Interested in developing games, optimizing code and technology in general.

Estimated contribution: 35 h
Absences: None

Contact information:
Email: tuomas.pekkanen@student.tut.fi

Markus Ylisiurunen, Scrum Master
Is familiar with C++, Python, JavaScript, PHP, Ruby. Has made a few side projects on his free time. Works as a software developer. Interested in the web as a whole.

Estimated contribution: 35 h
Absences: None

Contact information:
Email: markus.ylisiurunen@student.tut.fi

Jere Metsäranta, Product Owner
Prior experience limited to C++ and Python. Interested in learning new stuff, also majors in programming. Has one website project going on free time.

Estimated contribution: 35 h
Absences: None

Contact information:
Email: jere.metsaranta@student.tut.fi

Pedram Ghazi, Team Member
My preferred language for coding is Python but I also know C++, Matlab and PHP. I am also familiar with JavaScript. Right now, I am working part-time as a RA and field of my work is machine learning. Unfortunately, I do not have prior experience in C# or Unity but I assume the project of the course a good opportunity to learn these skills. In general, I am interested in AI and also web development.

Estimated contribution: 35 h
Absences: None

Contact information:
Email: pedram.ghazi@student.tut.fi

1.2 Process description

We had our first meeting at 16.1.2018 and we decided to create a Slack team where our group can communicate, ask questions and manage the project. We plan on relying on online communication for the majority of our communication needs. Whenever the need arises, we can meet face to face and discuss the current issues as a group. We have agreed to have a weekly online meeting at 18:00 every Tuesday.

At the very beginning of each sprint we will decide which features belong to the next sprint as a group. After that we'll assign those tasks equally to each member and everyone can work on their own time. This hopefully ensures that everyone will find the best time to work and it won't have too big of an impact on other things. Since we are using Slack as our communication method, everyone can ask questions at any given time.

Our goal is to split the tasks and work hours equally on each sprint and finishing sprints on the anticipated time. We want to maintain high quality code which has went through a code review. We also want to have a high visibility of the changes so that every group member has a good idea of what is currently being implemented and what is yet to be done. We also want to adapt ourselves to the found issues and make changes to our workflow as the project advances.

Our project goal is finishing on time and getting a high grade which also means that we are aiming for a good workflow, communication and high-quality code. We will meet this goal by planning in advance, having good communication and clear roles in the project.

1.3 Tools and technologies

We've decided to build the game in Unity. We are going to use the latest version (2017.3) and stay with that to prevent any issues from using different versions. Tuomas Pekkanen has previous experience in Unity and he'll be the person other group members can ask Unity related questions from.

We are using a repository hosted in GitLab to store our project. Each group member has access and can push new code. We'll utilize GitLab Merge Requests for code review. Each new feature/bug fix will go through a code review where at least one other group member will review the changes before merging to master. This way everyone can keep up with the changes and we hope to achieve better code quality.

Here are the relevant resources related to our project.

Repository: https://course-gitlab.tut.fi/sweng_2018/g06---ylo
AgileFant: https://app.agilefant.com/TTY-TIE/product/376284/tree

Table 1.1: Tools used in the project.

Purpose	**Tool**	**Contact person**	**version**
Communication	Slack https://slack.com/	M. Y.	-
Version management	Gitlab https://gitlab.com/	M. Y.	-
Development	Unity https://unity3d.com/	T. P.	2017.3

2. STUDY DIARY

This chapter holds your journal of lessons learned during the course. That is, **more detailed analysis of previous Sprint's contents**.

2.1 Sprint 1 (every sprint as a section)

2.1.1 What went well

2.1.2 What difficulties you had

2.1.3 What were the main learnings

2.1.4 What did you decide to change for the next sprint

2.2 Sprint 2

2.2.1 What went well

2.2.2 What difficulties you had

2.2.3 What were the main learnings

2.2.4 What did you decide to change for the next sprint

3. RISK MANAGEMENT PLAN

Consider risks for your project. **The most usual risks** that will affect projects are due to customer, the team itself and technology.

Just listing some risks at the beginning of the project doesn't help you much… if anything at all.

You can try to come up with **Plan Bs** for the risks. However, remember that the things you won't expect, will hurt you the most. Thus, focus on the generalities, not on specifics.

Try not to underestimate the probability of small and common risks, and not to overestimate the probability of rare and remarkable events. For example, people usually get 1-2 flus during a year, so in 4 months, it is quite probable that one of the team will be sick and may infect others, too. An average flu lasts for more than one week. So, be prepared. On the other hand, getting hurt in traffic so that it will take a week to recover happens to only for 15000 people yearly in Finland (less than 3 permille of population).
Be sensitive for weak signals, such as difficulties with new technology or runny noses.

You should think of risks in all categories:

- customer (ending the project, changing requirements, requirements remain unclear,…)
- technologies (hw/sw; hard to acquire, learning new technologies takes time, suitable library is not found,…)
- environment (network connections and servers fail,…)
- personnel (getting ill, changing jobs, busy with work,…)
- project management (bad scheduling, bad communication, forgetting things,…).

Usually we calculate risk's **seriousness = severity * probability**.

Table 4.1: [example] Project risks.

Risk ID	Description	Probab ility	Impa ct
P1	Short term absence	3	2
T1	Hard disk failure	2	2
	etc. …		

3.1 [example] Personnel risks

Try to estimate risk probability, use a scale of **1 to 3** (or 1..5) or Small, Medium, Large.
Other criterion will be the impact or severity. So, how the risk will harm you, if realized. Use similar scaling as in probability.

3.1.1 [example] Risk P1: short term absence of one person

Every major risk in the table will be further elaborated here. Analyze the risks, so that those risks which will hurt you the most are analyzed in more detail than rare and low-impact risks.
However, remember that the low impact risks may have cumulative effects, if they have high probability, and thus occur frequently.
Incorporate your mitigation methods to your process (see 1.2.). However, consider the sensibleness of the measures (risk severity vs. cost). For example, getting a flu shot (vaccination) for everyone in the team would surely be overkill.

Root cause (source): description of the risk. A key person will be absent for several days.
Importance (seriousness): from the table, basically probability and impact, possibly combined with frequency.
Avoidance: if you can lower the probability by preventive means, or even totally suppress (reject) the risk. For example, getting flu shots for everyone will lower the risk of short term sickness.
Response (prevention): means to take, if you have weak signals of looming disaster. For example, someone seems to be getting sick or will have a mandatory absence next week, redistribute the work load and share all relevant information, so that the team will be able to carry on.

Recovery (survival): the means to take, if other means have failed, and the risk has realized. Plan B. For example, redistribute the workload; focus on the most important features.

3.2 [example] Technology risks

3.2.1 [example] Risk T1: hard disk failure

Symptom, early warning sign: disk makes noise, arbitrary reading errors occur more often than before.
Source or reason: hard disk is at the end of its lifespan, or hard hit on computer while disk was running.
Probability: 2 medium (on scale 1-3)
Seriousness: 2 medium (on scale 1-3)
How to avoid: buy a new disk when starting a project.
How to prevent: when first symptoms occur, take additional back-ups and change the disk as soon as possible.
How to survive: back-ups, and a replacement disk or whole computer.