School of Electronics and Computer Science
Faculty of Engineering, Science and Mathematics
University of Southampton

Aikaterini Gkritsi

06/05/2010

Agile Game: A Project Management Game for Agile Methods

Project supervisor: Dr Andrew M Gravell
Second examiner: Ed J Zaluska

A project progress report submitted for the award of
BEng Software Engineering

Abstract

Since mid-1990s, companies have adopted agile methods and incorporated them in their development methodologies. For this reason, future project managers and developers need to have a full understanding of these methods. At present, the university's approach to agile methods is theoretical and is not reflected during the development of a product and their practical use. The purpose of this project is the creation of a software system in the form of a game, named Agile Game, which simulates their use. The system is designed for use as supplementary material in lectures, to help students understand agile methods, to present their use within a project, and to demonstrate how they differ from traditional project management methodologies. The final system, which is web based, was implemented using PHP, MySQL and JavaScript. It was fully tested against the requirements and evaluated by peer students. The evaluation showed that the majority of users were satisfied with the system but they thought that it should contain more detailed information at every step of the game. For this reason, some parts of the design and the content were reviewed to meet user requirements.

Acknowledgements

First of all, I would like to say a very big thank you my supervisor Dr Andy Gravell who despite all the difficulties and problems that occurred during the project he was always there for me, happy to help, advice and guide me. Also, I would like to thank Mr Quintin Gee for his invaluable help proof-reading this report. Finally, I would also like to thank my friend George for his advice on many aspects of my project as well as all the people that participated in my survey and interview because without them I would not be able to complete this project.

Table of Contents

1 Introduction

Agile methods are project management processes that allow a more dynamic and flexible management of a project compared with traditional project management methodologies. Because of these features, businesses have adopted agile methods and successfully applied them to projects of different sizes and tailored them to the needs of each project team. In 2008, Scott Ambler carried out a survey concerning the level of adoption of agile methods within businesses. The results showed that 69% of the responders work for companies that have already adopted at least one agile method and 15% are employed by businesses that plan to 'become agile' within a year (Ambler 2008). Since agile methods play an important role in project management and many businesses, the developers and project managers in these companies need to have a full understanding of these methods and concepts.

This knowledge and understanding can be gained from work experience but the foundations are built whilst at university. At present, the Southampton University's approach to agile methods is theoretical and is not reflected in their impact during the development of a product and their practical use, so students do not have a clear view of their application. From the above statistics, it seems crucial that students become equally familiar with agile as with traditional project management methods before leaving university. Because some students may not have previous work experience, the theoretical knowledge provided by the lectures can be enriched by combining them with a software system that simulates the use of agile methods. This way, students will be able to get a glimpse and gain deeper understanding during the learning process of what agile methods are and how they are used by businesses.

It has been shown that students who used additional software, in the form of games in parallel with lectures, performed better on their final examinations compared with others that attended only the lectures during their course (Clua et al. 2006). A system in the form of a game can be a good option of helping students learn more details about agile methods. The aim of this project was to design and implement a game prototype that simulates all these: the Agile Game. The Agile Game was designed to be used as part of additional teaching material at the university and, besides introducing students to software development and agile methods, allow them to grow their learning and analytical skills.

1.1 Project Description

The Agile Game is a prototype of a game that simulates the use and the impact of agile methods on every phase in the development of a project. The system is targeted at university students with some background knowledge of software engineering issues. Because of this, the users of the game need to be students in Computer Science, Software Engineering or in other IT-related degrees. In addition, because some knowledge of software engineering is assumed, the game is more suitable for students that have already completed their first year at university. Through this game, users will come to understand the phases that a project has to undergo for an enterprise to deliver a product to the customer, and how agile methods differ from traditional project

management. The key aspect of this system is its form. Educational software in the form of a game will trigger students' attention, develop their understanding, motivate them and will make the learning process more interesting and challenging (Basturk 2005). Clua et al. characteristically state that "much research shows that the learning process is highly enhanced when this kind of approach is used in computer science teaching, not only because of the motivation they engender but also because high end results can be easily generated with relatively little effort" (Clua et al. 2006).

1.1.1 Detailed Description

When starting to play the game, users are required to register with the system by inserting a username and a password. The password is stored as a hash in the database to ensure security. When a player first logs in, they are given instructions on how to play the game and a detailed project profile containing information about the project they are required to complete. The game simulates a software project that is divided into four smaller phases (User Stories/Requirements, Design/Planning, Implementation, and Acceptance Testing). Each of these represents the basic phases that a project has to undergo until its final delivery. In the game, users will have the role of Project Manager, and will be responsible for taking all the necessary decisions for the completion of every deliverable and the management of a virtual team. Taking into consideration the data of the project profile provided, users will have to decide which methods and techniques would be more appropriate for each deliverable.

Every choice is credited with a certain number of points, depending on the phase of the project and many other factors. The system will keep track of the points that every player collects. These points will be visible to all other players in the system in the form of a high score table. With this feature, players will be able to compare their performance with respect to other players in each deliverable and in the overall project. The score table makes the game more interesting and keeps the player's motivation high, thus challenging them to perform better in every deliverable.

Before starting and after completing the game, players are required to fill in a questionnaire. The first questionnaire contains simple questions testing basic knowledge on agile methods. Each answer is rewarded with a number of points. If the final score is above a certain limit, users can proceed to the first phase of the game. Otherwise, users are recommended to consult the additional resources provided by the help webpage. The second questionnaire, at the end of the game, contains more detailed questions concerning the techniques that players used during the game. Again, each answer corresponds to a number of points. Before exiting the game, users can see their scores in both questionnaires and track their progress.

2 Background research and literature search

This project focuses on the use of project management methodologies and it was developed using the principles of Extreme Programming (XP) (Section 2.3.1). As required by the structure of XP, the planning and the iteration phase of the project cannot start without first defining the user requirements. To clarify the requirements of the system, it was essential to undertake a literature review and background research on existing and related work. The following section is a summarised review of the literature, but a more detailed review can be found in Appendix A.

2.1 Outline

The first step of the research was to understand in more depth the term *project management* and make clear the key characteristics that distinguish traditional project management methodologies (Section 2.2) from agile methods (Section 2.3). Then, further research was conducted on the methods that are considered as agile, such as XP, Scrum, Crystal Clear, Dynamic Systems Development Method (DSDM) and Feature Driven Development (FDD), which are considered to be the most popular (Ambler 2006 Parsons et al. 2007; Lindvall 2002) and widely used methods (Section 2.3). The investigation addressed at what level businesses use these methods (Appendix A), for what kind of system (critical or non-critical projects) and what techniques are used by each method. Also investigated were the level of adoption of agile methods by businesses and the affect on total cost of the project using agile methods compared with traditional methods (Appendix A). It was also important to see how they influence the productivity of the team, the quality of the final deliverable and how satisfied customers are when projects are developed with agile methods (Appendix A).

Unfortunately, it was not possible to find specific information and statistics about which methods are used in each phase of a project, and which techniques are used alone or in combination with others during different moments in the project lifecycle. Also, there was limited data describing the phases of the project in which the risk of failure is high using agile methods compared with traditional methods, which methods tend to have a lower success rate than others, and what companies do to avoid these risks or limit their consequences. Moreover, it was not possible to determine in what types of project agile methods are usually applied. There are probably two reasons for these limitations. First, agile methods are relatively new in project management and there has not been enough time to assess their overall and long-term effectiveness. Secondly, this kind of information is often a business secret so publishing this information would jeopardise the advantage of these businesses.

Work previously done in this field was investigated (Section 2.4) in order to assess what features the existing systems have and in what way the Agile Game could differ (Section 2.4.2). Because this system has the form of a game, some research was conducted to determine whether educational systems like Agile Game are effective, assist the learning process and enhance students' understanding (Section 2.5). Finally, for the purposes of this system and the project report, some background research was necessary in order to decide whether the requirements of the project would have the form of Use cases or user stories (Appendix A), to determine the issues that a risk analysis needed to include, and the way that Gantt charts are constructed.

2.2 Principles of traditional project management

The Project Management Body of Knowledge (PMBOK) states that project management is "the application of knowledge, skills, tools and techniques to project activities to achieve requirements. Project management is accomplished through the application and integration of the project management processes of initiating, planning, executing, monitoring and controlling and closing" (Lewis 2007).

The term *traditional project management* refers to software models that focus on the plan of the project, analysis, design, and quality assurance. One of the most popular models of this kind is the waterfall model (Figure 2.1).

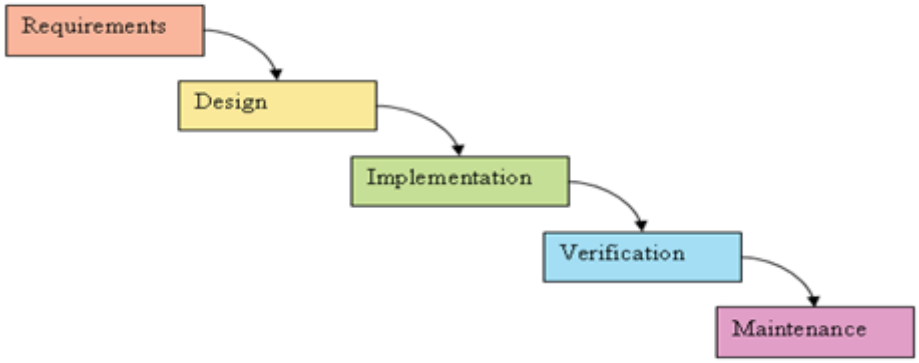

Figure 2.1: The waterfall model diagram (Serena 2007)

This model takes the fundamental processes of specification, development, validation and evolution and represents them as phases of a project, i.e. requirements, design, implementation, verification and maintenance (Somerville 2001). Some other traditional project management models are the Spiral model and the V-Model (Appendix A). This type of model is usually adopted by large businesses with big teams who are responsible of projects with long duration. As Somerville says, when smaller companies apply these models, they are dominated by the software development process (Somerville 2007). For this reason, businesses developed agile methods and introduced them into project management.

2.3 Agile methods

Agile methods allow a more dynamic and interactive development of a project than traditional project management. They were first used by medium- and small-sized businesses because they could not afford the heavyweight approach of traditional project management that large businesses were using. Agile methods are characterised by their incremental delivery and development of projects. "These allowed the development team to focus on the software itself rather than on its design and documentation" (Somerville 2007). The main difference between agile methods and traditional project management is that the design and the requirements can change at any time, contrasted with models like Waterfall, where a design is completely developed first so the product is then designed, implemented and tested against that initial design (Aguanno 2004). The difference also becomes clear from the *Manifesto*

for Agile Software Development[1] which places "individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation and finally responding to change over following a plan" (Beck 2000). Some of the commonly used agile methods are Extreme Programming, Scrum, Feature Driven Development, Dynamic Systems Development Method, and Crystal Clear (Ambler 2006; Parsons et al. 2007; Lindvall 2002).

2.3.1 Extreme Programming (XP)

"XP is a path of improvement to excellence for people coming together to develop software" (Beck et al. 2004). This method focuses more on the implementation than the documentation of the project and emphasises the customer involvement and testing. In XP, the user requirements are expressed as user stories or scenarios from the customer/ stakeholder (Figure 2.2). This list of features forms the release plan of the project that indicates which stories will be implemented first and in which iteration. Each iteration has a relatively short duration (usually 2-4 weeks) and always needs to deliver some functionality after its completion. The releases need to be small but simple in order to get frequent and precise feedback from the customer, which is very helpful, especially for large projects (Highsmith 2002). As Kent Beck characteristically mentions, "Every release should be as small as possible, containing the most valuable business requirements" (Beck 2000). In addition, the design of the system must confront the given specifications and not consider possible future enhancements. In XP the team needs to do only what is specified, but in the most effective and productive way (McDonald [n.d.] a). Moreover, as soon as the release plan is conducted, during the iteration planning, teams create acceptance tests based on the requirements and check the functionality of the deliverable. Acceptance tests are another way to describe black box testing and each test corresponds only to one user story (Wells 1997-1999).

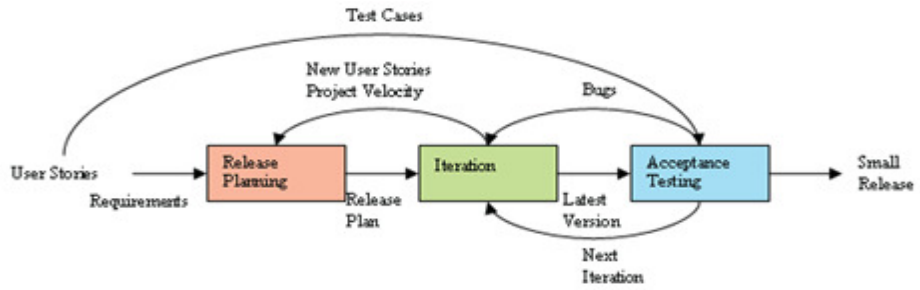

Figure 2.2: XP model diagram (Serena 2007)

XP is typically used by small teams of at most 10 people, which are co-located, although Martin Fowler tried to use it with larger teams (about 40 people) with satisfying results in terms of understanding and planning (Highsmith 2002). A co-located team is one whose members work either in the same room, or on the same floor, or in the same building, thus reinforcing teamwork, communication and collaboration. A typical application of XP was the Chrysler Comprehensive Compensation project (mid-1990s). The project was not completed, but it was partly implemented and it proved that XP methodology can be used as a development method (Highsmith 2002).

[1] Available from: http://agilemanifesto.org/

This method is a collection of good engineering practices (McDonald [n.d.] a). Some of the most common techniques that they use are pair programming and refactoring. In pair programming, programmers work in pairs so they develop code efficiently and with higher quality. A survey by the University of Utah showed that the use of pair programming while developing software helps in faster delivery and higher quality (Williams et al. 2000). Code refactoring is the change to the code of an existing software system, without changing its external functionality (Fowler et al. 2004). This technique is used to improve the reliability and reduce the complexity of the system. It also improves maintainability, extensibility and regular testing before new code is integrated into the system (Wake 2001), which helps to minimise the number of bugs in the system.

2.3.2 Scrum

In Scrum, the control is moved from the central scheduling and dispatch authority to the individual teams (Schwaber 2004). Jim Highsmith notes that "whereas XP has a definite programming flavour (pair programming, coding standards, refactoring), Scrum has a project management emphasis" (Highsmith 2002). The product of every iteration in Scrum is an increment of the final product. This agile method is considered easy to learn and it does not need much effort to start using it (Henson 2008).

In Scrum, Product Owners maintain a Product Backlog (Figure 2.3) which contains all the features that they want the system to include, and they prioritise them. Then again, the Product Owners choose the features they want to be released in the next iteration (Release Backlog). In a planning meeting the Product Owners, the management and the team, estimate the amount of work that is required to complete these tasks (Sprint Backlog). The development period of Scrum is divided into 2-4 week iterations, called Sprints. During Sprints, the team needs to participate in daily Scrum Meetings in order to identify the problems that the members of the team might face and find ways to resolve them. Daily Scrum Meetings take place every day, usually at the same place and they last less than 30 minutes (the ideal is 15 minutes). They are held by the Scrum Master who is responsible for identifying the team's problems, and monitoring their overall progress. To identify the possible obstacles and difficulties faced by a member and their progress during this meeting, every member of the team has to answer the following three questions:

- What have you done since last meeting?
- What will you do now and for the next meeting?
- What problems do you have? (Highsmith 2002; Schwaber 2004).

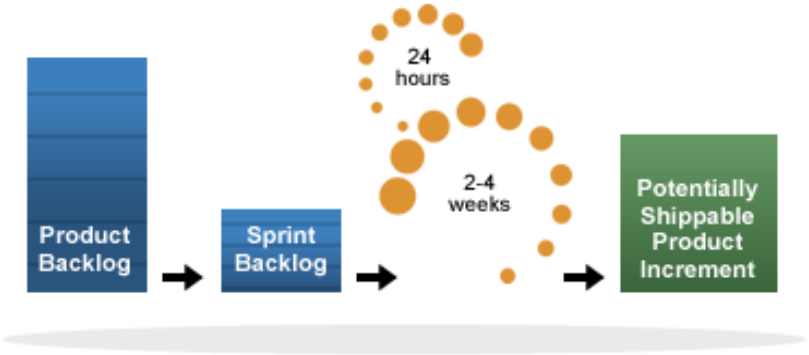

Figure 2.3: Scrum method diagram (Scrum Alliance 2009)

As soon as a Sprint is complete, a 4-hour Sprint review meeting takes place in order to evaluate the new features of the product against the Sprint goals, to monitor the overall progress of the product and to present the new features of the system to the Product Owners. In the development phase, the team uses a Burndown chart to monitor the amount of work that still needs to be done. With this chart, it is possible to track the proportion of the remaining work and the effort that is needed to reduce this workload. The point where the trend line of the graph intercepts the horizontal axis of the graph represents the time that the project is expected to finish (Highsmith 2002).

2.3.3 Feature Driven Development (FDD)

FDD focuses on the design of the project and not on its development. This method is characterised by its interactive development, incremental delivery and emphasis on quality (Abrahamsson 2002). It includes some prescription about what the tasks are and who is responsible for these tasks, so many do not consider it a truly agile method. It is considered good for companies that are changing from traditional to iterative approach but are not comfortable with getting rid of all the tasks and assignments (McDonald [n.d.] b).

A popular example of this method is the commercial lending application project for a large bank in Singapore. The company that was first assigned to implement it spent two years delivering thousands of pages of Use cases and object models, but without any code. Then, with Jeff De Luca (architect of FDD) as the project manager, the project was implemented using FDD in a period of 15 months, with about 2000 features delivered. “The key, Jeff De Luca said, is having good people – good domain experts, good developers and good chief programmers” (Highsmith 2002).

FFD lifecycle is divided into five different activities that are performed interactively (Figure 2.4) and must be short, iterative and feature driven. An FDD project starts by performing the first three steps. The goal of the project is to identify the amount of effort, the initial architecture, and plan. Construction efforts occur in two-week (or shorter) iterations, with teams working iteratively throughout the five steps as needed (Ambler 2005-2009).

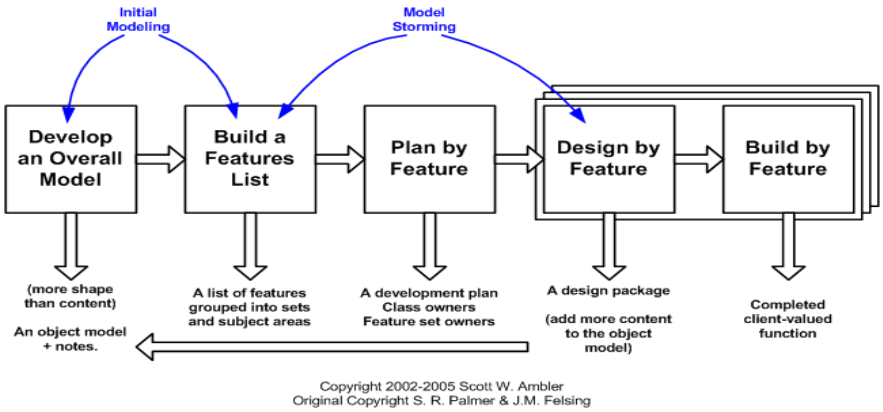

Figure 2.4: FDD Model diagram (Ambler 2005-2009)

The five different processes of FDD are Develop an Overall Model, Build a Features List, Plan by Feature, and Design and Build by Feature. In particular, the first process is the stage where the domain and the scope of the project are defined. As soon as the Use cases are complete, they are used to create the overall model and are then integrated into

features in the next process. In the Features List process, the team develops a list of features, groups them into feature sets and finally into major feature sets. Every feature must be completed within 10 days. If a feature is expected to last more than 10 days, it needs to be divided into smaller pieces. In the Plan by Feature process, the team along with the project manager, the development manager and the chief programmers, construct a plan for the development phase that defines the features that will be implemented and the people responsible for their completion. Finally, the last two processes are where the team performs multiple iterations of these processes; they break into Feature teams and implement classes and methods, inspect code and perform unit testing in two week time-boxes (Highsmith 2002).

2.3.4 Dynamic Systems Development Method (DSDM)

XP is considered one of the first well-known methods to handle agile software projects and it can be integrated into DSDM implementation because its principles can improve XP with more robust requirements and project management mechanisms (Voigt 2004). DSDM supports the notion that nothing is built the best possible way the first time (Highsmith 2002). The DSDM method follows 9 principles. It does not force its users to follow its complete structure, but only requires strictly following these 9 principles. If it is not possible to implement all of the 9 principles, then DSDM is not the most suitable method to implement a project. These principles are:

1. Active stakeholder participation.
2. Teams empowered to make decisions.
3. Focus on frequent delivery.
4. Use fitness for business purpose as criterion for accepted deliverables.
5. Iterative and Incremental development is essential.
6. Changes during the development phase must be reversible.
7. Requirements base-lined at a high level.
8. Continuous integrated testing.
9. Collaboration and cooperation between all stakeholders.

DSSM emphasises facilitated workshops as well as customer and user involvement, so DSDM design is done with respect to their needs and expectations. Projects that are implemented using the DSDM method consist of several phases of which some might be omitted to tailor the method to the needs of each project (Figure 2.5) (Highsmith 2002; Voigt 2004).

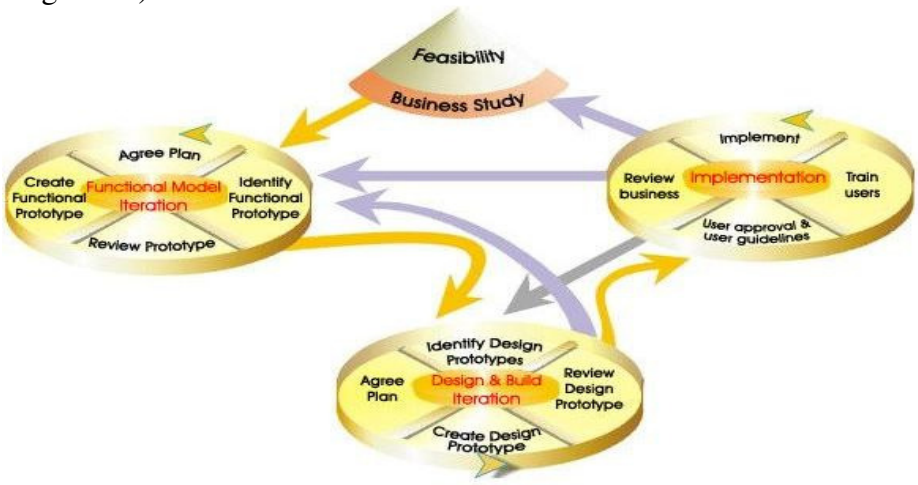

Figure 2.5: DSDM method diagram (Voigt 2004)

Some major phases are:

1. Feasibility & Business Study: In this phase the problem, and assessments of likely costs and technical feasibility to deliver the product, are defined and the business study provides the basis for all subsequent work. It is as short as possible, while achieving sufficient understanding of the requirements.
2. Functional Model Interaction: The focus of this phase is to refine the business-based aspects of the computer system. The functional and non-functional requirements are defined and prioritised and they are usually represented in the form of prototypes, rather than text.
3. Design and Build Iteration: In this phase, the system is engineered to a high standard to be safely delivered to the user. In addition, the prototypes from the previous process are checked against the user requirements.
4. Implementation: This phase is the transition from the development to the operational environment (Highsmith 2002; Voigt 2004).

2.3.5 Crystal Clear

Crystal is a human powered and adaptive agile method. It achieves the project success through developing the work of the people involved. Crystal is a family of methodologies comparable to other agile methods (Cockburn 2009). Crystal Clear is actually one of the four methodologies of Crystal: Crystal Yellow, Crystal Orange and Crystal Red (Figure 2.6). Crystal has the ability to tailor these methodologies to the needs of each project and uses incremental development cycles with a maximum duration of four months (Abrahamsson 2002).

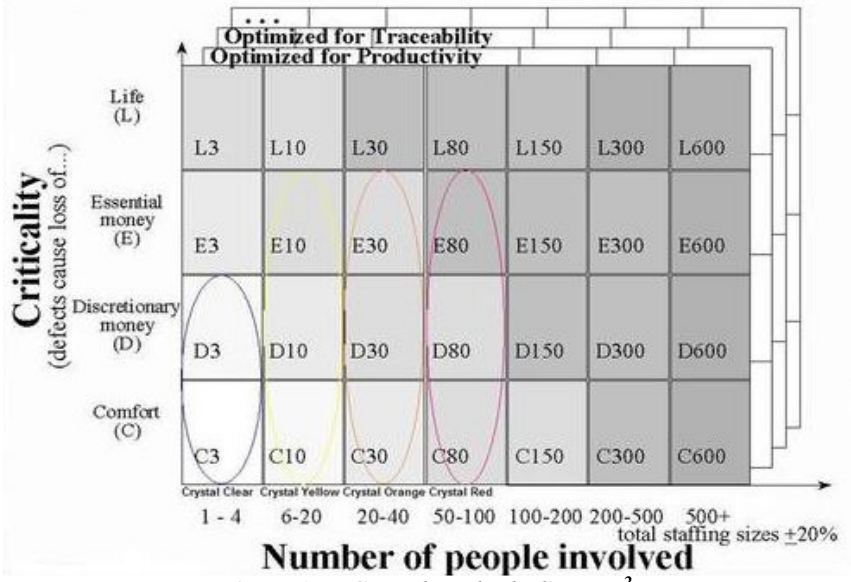

Figure 2.6: Crystal methods diagram[2]

As Highsmith states, Crystal "focuses on people, interaction, community, skills, talents and communications, as first-order effects on performance. Process remains important, but secondary" (Highsmith 2002). This method supports the notion that because each person has their own talents and strengths, they should be assigned to tasks that match their skills. It is considered as a human powered, ultra-light and stretch-to-fit method.

[2] Available from: http://leadinganswers.typepad.com/leading_answers/images/2007/06/20/3_crystal.jpg

This is because it prioritises people before work, supports a minimum of documentation, and can be adjusted to the needs of any project at any time (Chang 2010).

The method chosen from the Crystal family that is more applicable to a project depends on its size and how critical it is, i.e. loss of comfort, loss of money or death. Crystal Clear in particular, is used in small teams of up to six people and on projects with low criticality (Chang 2010). In this method, if the team is not co-located they cannot communicate. A Crystal Clear team consists of a Project Coordinator, a Business Expert, some Requirements Gatherers and finally a Senior Programmer. The development with Crystal Clear is divided into various iterations each lasting 2-3 months so the final product can be incrementally delivered. The progress of the overall project can be monitored by milestones that represent every deliverable. In addition, to test the code, some regression and user (usually two people) testing takes place. Finally, a project developed with XP can also be implemented with Crystal because the latter fulfils all the XP standards, except documentation (Cockburn 2002).

2.3.6 Agile Methods within the Agile Game

The principles of all the previously mentioned methods are the core of the Agile Game. As can be seen above, there are several agile methods and each one has its own unique principles and techniques. The aim of this system is to familiarise students with agile methods and introduce them to these techniques and principles during the game, so detailed information about them and their use is provided to fulfil this goal.

2.4 Existing Games

During the research phase, to specify the requirements of this project, it was necessary to investigate what kind of systems already exist and which of their features the Agile Game could adopt or enhance and what functionality would differentiate it from the existing ones.

2.4.1 Overview

It was found that many games are used in education and help to enrich the teaching process. Some of the most characteristic games are: “The Software Management Game”, “The Agile Hour”, “The XP Game”, “Contract & Construct (C & C)” and “The Incredible Manager”.

In particular, the game “Software Management Game” was developed by Dr P W Garratt from the University of Southampton, and it simulates a computerised information system. Through this game, users have the opportunity to familiarise themselves with the principles of traditional project management and enrich their communication and negotiation skills. The game is a complete system that simulates accurately the phases that a project has to undergo to be delivered to the customer. In the game, users may occupy different managerial positions so they can view different perspectives the project development, the responsibilities of a person in that position and the task relevant to their position (Garratt 1995). Likewise, the “Agile Hour” is another project management game, which is a simulation game that focuses on agile methods and, in particular, the XP methodology and its techniques. During the game,

players use story cards and are required to build a human-powered vehicle. The game is not computer-based, so the team actually needs to gather in the same place in order to communicate and coordinate. Users may hold different roles and, due to the iterative and incremental nature of XP programming, they can change, add or remove techniques during the project implementation (Parsons & Cranshaw 2008).

The "XP Game" is again a story card-based simulation game, where no technical knowledge or skills are required. In the game users are divided into teams of developers and business people. The goal of the game is to experience the way that user stories, estimation, planning, implementation and functional tests are used. Players use cards that contain simple tasks, which correspond to a score. The game has at least three iterations and in each iteration, teams perform a planning game session, which is based on the story cards (Peeters et al. 2008). In addition, Contract & Construct (which is an implementation of the Project Management Simulation Engine) is an educational game designed by the Business School of Warwick to support the teaching of project management for an MBA. The game simulates "all the classical functional management elements of planning, command, co-ordination and control" (Martin 2000). Users are given a detailed project description, the events that might occur during the development of the project and their impact, and the budget, requirements and constraints of the stakeholder. This game focuses more on decision-making for general issues during the development rather than specifically on which project management method should be used (Martin 2000).

Lastly, the "Incredible Manager" is again a simulation software-based game aimed at students, but it can also be used to provide experimental learning for project managers. As Dantas et al. state, the system can be "used for educational goals, aiming at reasoning, judgement, decision-making and system thinking." It is also used as additional material in teaching. The different characters in the game allow students to understand the responsibilities of every position and the different phases of the project, throughout the lifecycle of a real project (Dantas 2004).

2.4.2 Comparison of existing games

The table below (Table 2.1) represents the main features of the existing games along with the features of the Agile Game. The majority of the systems are used as supplementary teaching material to enrich the technical experience of the students but some of them are designed as complete and stand alone systems, which means that their users do not need to have any prior knowledge on the field to play. The project management issues that each game covers vary. In particular, they either represent the application of agile methods (like Agile Hour, XP Game and Agile Game), the principles of traditional project management (Software Management Game) or cover general managerial decisions (e.g. what be the most appropriate next move given a situation, which people are more appropriate in a position than others) which do not focus on the technical aspects of project management i.e. which method would be more appropriate in a particular phase of the project. Finally, half of these games are computer based, so their users can use them at any time without dedicating large amounts of time at once (whereas in XP Game a project lifecycle lasts about 40 minutes and in Agile Hour 70 minutes).

Game Type	Agile Hour	Incredible Manager	Contract & Construct	Software Management Game	XP Game	Agile Game
Educational game		✓	✓	✓		✓
Independent system	✓		✓		✓	✓
Traditional methods				✓		
Agile methods	✓				✓	✓
General management		✓	✓			
Computer based game		✓	✓			✓
Interpersonal game	✓			✓	✓	

Table 2.1: Existing games comparison

2.4.3 Why Agile Game?

Comparing the Agile Game with existing games, it is easy to note that this system contains many features similar to all the other systems. Specifically, Agile Game is an educational system designed to be used as supplementary material in teaching. It is also computer-based, a feature that makes it accessible to the majority of its targeted users (i.e. ECS students). The main difference between the Agile Game and other systems is that it is web-based which means that players are able to use it even if they do not have access to their personal computer.

The most important feature that distinguishes the Agile Game from similar games is that it teaches and helps students understand a variety of agile methods and their techniques. In particular, unlike Agile Hour and XP Game which focus only on XP techniques and XP methodology respectively, the Agile Game offers its users the opportunity to familiarise themselves with some of the most commonly used agile methods, as well as with their techniques. In the Agile Game, users are able to understand for every method, which technique is most appropriate, depending on the phase of the project and the reason why this happens.

2.5 Learning through Computer Games

The use of computer games in education is a controversial issue because there are doubts whether students learn through such a means. Because the Agile Game is an educational game, it was necessary to investigate if the form of the system would actually help students to learn. Lots of research has been conducted on the subject and the results seem very positive, and there are many examples of universities around the world that have already incorporated educational games in their teaching material. A survey from the University of North Carolina concerning a game for first year computer scientists, which teaches programming, revealed that 88% of the students would use the game as additional material (Barnes et al. 2008). There were also cases where students

that used educational games in combination with the teaching material improved their overall performance, and their motivation of learning was raised (Virvou et al. 2005).

Educational games are usually confused with the video games that students play for amusement. As Becker says "the vast majority of the educational software available today is presented in the form of games of one sort or another" (Becker 2001). Unlike video games that are considered as action or fighting games, there are video games in the form of simulations, strategy games, role playing, sports, etc. These types of game, like simulation games, represent a model of the world that is very close to reality. This model is usually abstract or simplified for the purposes of the game, but they do not suspend the rules of reality as action games do (Squire 2003). Galvão et al. state that "simulation games are a mixed feature of a game competition, co-operation, participants and rules incorporating critical features of reality." They continue that educational games need to create awareness and insight for the student while teaching them (Galvão et al. 2000). Furthermore, Oblinger believes that a game that is educational needs to follow some general guidelines, which are described in the following table (Oblinger 2004).

Principle	Description	Application in Games
Individualization	Learning is tailored to the needs of the individual	Games adapt to the level of the individual
Feedback	Immediate and contextual feedback improves learning and reduces uncertainty	Games provide immediate and contextualized feedback
Active Learning	Learning should engage the learner in active discovery and construction of new knowledge	Games provide an active environment which leads to discovery
Motivation	Students are motivated when presented with meaningful and rewarding activities	Games engage users for hours of engagement in pursuit of a goal
Social	Learning is a social and participatory process	Games can be played with others or involve communities of users interested in the same game
Scaffolding	Leaders are gradually challenged with greater levels of difficulty in a progression that allows them to be successful in incremental steps	Games are built with multiple levels; players cannot move to a higher level until competence is displayed at the current level
Transfer	Learners develop the ability to transfer learning from one situation to another	Games allow users to transfer information from an existing context to a novel one
Assessment	Individuals have the opportunity to assess their own learning and/or compare it to that of others	Games allow users to evaluate their skill and compare themselves to others

Table 2.2: Some principles of good pedagogy and parallels in a game environment (Oblinger 2004)

There are many examples where universities have used educational games. For example, the University of Phoenix uses the "Thinking Strategically" simulation game in the MBA, Undergraduate and Business Management courses, which teaches the roles that a person can occupy within a company and what their responsibilities are (Oblinger 2004). The University of Piraeus in Greece used a virtual reality game, VR-Engage, to teach students geography (Virvou et al. 2005). The University of Michigan used the Conflix simulation game to allow students to discuss political and social issues, in order

to develop their analytical and negotiation skills (Oblinger 2004). The US Defence Intelligence Agency uses e-games to train their agents and soldiers simulating war situations under circumstances that would be dangerous and costly to set up in reality (Gotterbarn 2008). In Sweden, high schools use on-line learning games to teach different courses such as mathematics, physics, business administration. The University of North Carolina uses the interactive game "Game 2 Learn" to teach their first year computer science students programming (Barnes et al. 2008).

3 Project Management

3.1 Planning – Gantt chart

Cohn characteristically states that “estimating and planning are critical to the success of any software development project of any size or consequence” (Cohn 2007 a). The previous statement underlines the importance of planning in every project. In particular, plans work as a guide during the whole duration of the project to avoid losing focus, help knowing at which stage the project is at a specific time, and what needs to be done next. They also help to see if the project is within its time limits, keep track of the overall progress at any time and estimate the amount of work that still needs to be done until the final deliverable.

The initial plan of this project is shown below (Figure 3.1). The final Gantt chart (Figure B.6) and Gantt charts for every semester can be found in Appendix B. At the beginning of the project, a basic Gantt chart (Figure B.1) was constructed. It contained only the main phases of the project with very draft estimations because the requirements of the project were not fully specified. As soon as the precise subject of the project was defined and finalised, a more detailed Gantt chart was essential representing every task in the project. Figure B.2 represents the chart for Semester One, before the start of the project. However, the progress of the project did not go as planned and Figure B.3 shows how it was altered by the end of the semester. Figure B.4 represents the plan for the beginning of Semester Two and Figure B.5 the way that the project was formed at the end. Finally, because the project was developed with agile methods, it is focused on planning and not on the plan itself. Plans are flexible because the project has much iteration and its requirements constantly change so planning continues throughout the project (Cohn 2007 a).

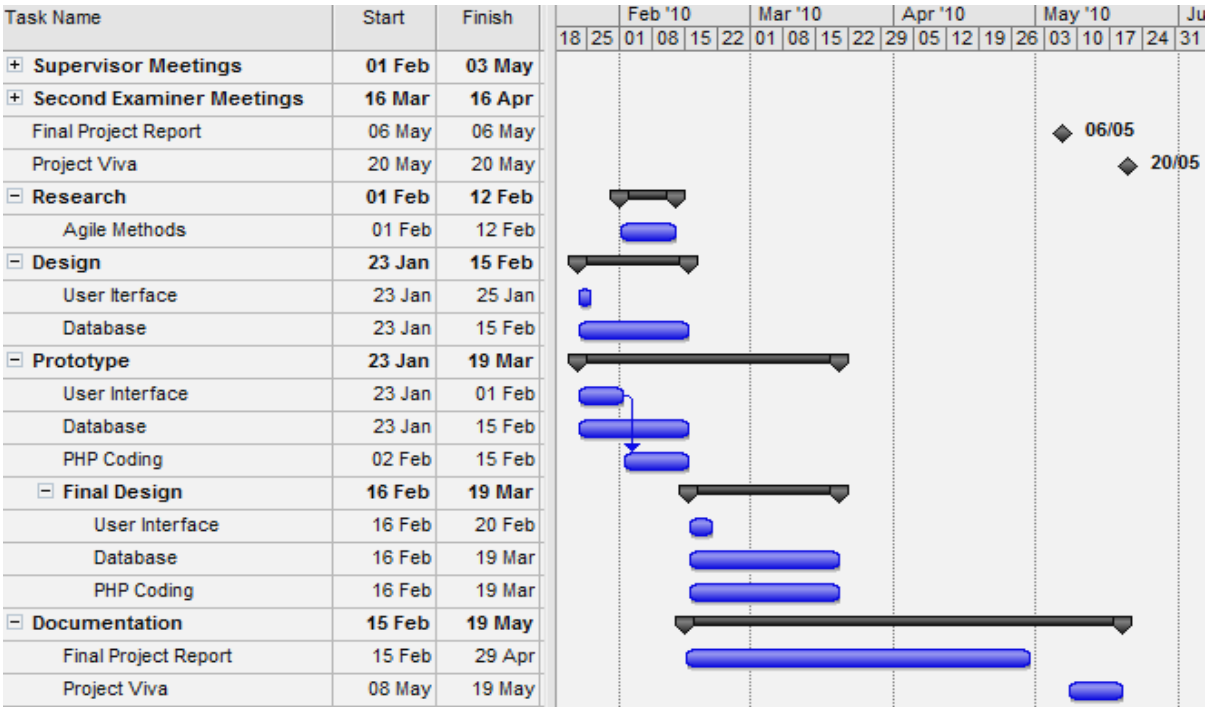

Figure 3.1: Initial Gantt chart

3.2 Risk Analysis

During the development of a project, many unexpected events might happen with consequences on the quality, budget, delivery or overall progress of a project. These events are usually caused by external sources so the time and the probability of their occurrence cannot be predicted. To recover from these events with minimum consequences, a risk analysis is necessary. Table 3.1 represents a number of events that might affect the progress of the project and their impact. The table contains an estimation of the probability of their occurrence and presents ways to deal with every risk to minimise their consequences.

	1=Very Unlikely; 2=Unlikely; 3=Moderate; 4=Likely		
Risk	**Impact**	**Probability**	**Strategy to deal with risk**
Illness	Unable to complete work and pending tasks are delayed.	1 (Serious) 3 (Non-Serious)	Reschedule tasks left to complete in the time available.
Database corruption	System unable to access the database.	1	Regularly back up the database and inform users of the situation.
Hard disk failure	A big part of the work may be lost.	3	Perform regular backups and save work into online repositories.
Missing deadlines	Project goes out of schedule and loss of marks.	4	Try to keep on schedule or reschedule tasks in the time available.
Missing supervisor meetings	Lose the focus of the project.	2	Inform Personal Tutor and get advice from him and the second examiner.
ECS filestore failure	System will not be accessible to third parties.	1	Keep local copies of the system.
Lack of players	Students may not test the system.	1	Change the target group of the system and find volunteers to use it.
Insecure system	System will be vulnerable to external attacks.	3	Implement security procedures.
Coding restrictions	Unable to complete desired features.	4	Find alternate ways to implement the feature; ask colleagues to help; advice from online sources.
Extreme weather conditions	Unable to be in England and submit the written version of the report.	1	Ask a colleague to submit the paper report instead.

Table 3.1: Risk Analysis

3.3 Project Methodology

The overall development of the project follows the structure of XP Programming methodology (Figure 2.2) with minor differences. This model was used because the project was programming-oriented and more focus needed to be drawn to the implementation rather than to other aspects. Also, programming languages were used in which the author had limited experience, so that reviewing the code as the project progressed would help optimising and refactoring as well as the database.

The XP lifecycle emphasises implementation and allows continuous iteration for the duration of the project so the system can be constantly reviewed and tested. The implementation of the code is usually done by a pair of programmers but it is not applicable in this case. In particular, the system had frequent releases with very short development cycles just like XP; progress of the project is presented every week in supervisor meetings. In addition, the design focused more on the development (continuous integration) of the game rather than the design and documentation, due to the short period within which the project needed to be implemented. Moreover, some code needed to be written every week to keep to schedule, so XP was considered to be the best method given these requirements.

Finally, the design fulfils the interactive approach of XP method because the requirements of the system change quickly during the implementation and black box testing regularly takes place to meet user stories.

4 Design

For the implementation of the game, a design was essential in order to use it as a guide. This design was the product of the requirements, the system and the project goals of the system that were set at the beginning of the project (Appendix C). During the implementation process, the requirements changed, therefore the system needed to be redesigned. This occurred because additional functionality was required to make the system more interactive and educational, and as a result of the evaluation questionnaire where users required some more detailed information. Because the project was implemented with XP methodology, the occurrence of changes was expected, so only a few modifications were needed (Appendix C). The initial and the final designs are described below, but more specific details such as the database diagrams can be found in Appendix D.

4.1 Initial Design

The aim of the initial design was to create a web-based game that simulates the principles of the XP method and represents how the choice of certain techniques affects the development process of the project and the final deliverable. The game was divided into four smaller deliverables (User Stories/Requirements, Design/Planning, Implementation, and Acceptance Testing) where each represents the basic phases that a project has to undergo until its final delivery.

4.1.1 System

Since the system was web based, users needed to interact with the internet. The PHP scripting language that was chosen for the implementation of the project offers advanced libraries and ease of integration. Because it is a server-side language, the users' browser cannot recognise the PHP files without accessing the server, which presents the PHP file in HTML format. This feature makes the code more secure compared with other scripting languages such as JavaScript, where the full code is visible to the user. All the necessary system information is stored in a database that can be accessed by queries from the PHP files. Figure 4.1 represents the interaction between the user and the system.

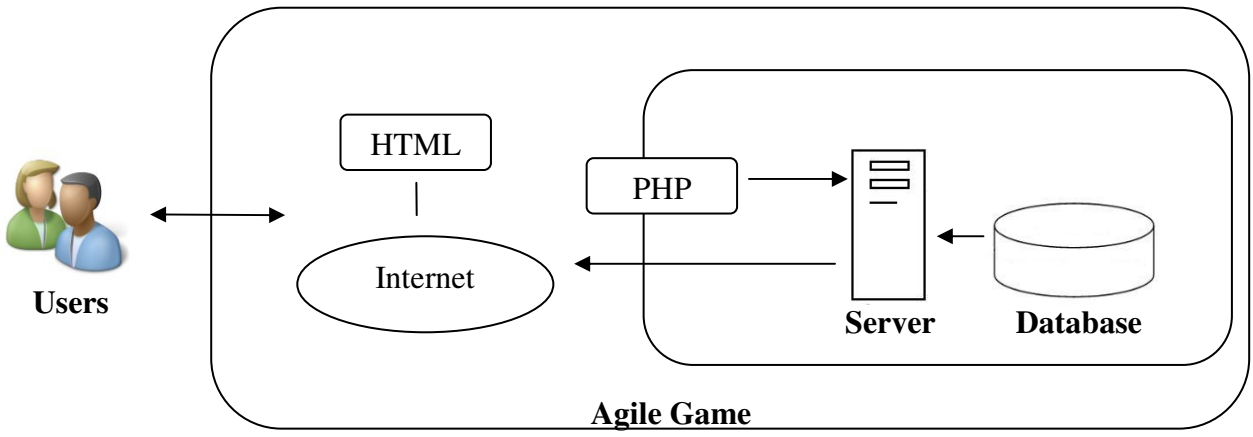

Figure 4.1: System diagram

4.1.2 Initial Database

The complete model of the database of the initial design can be found in Appendix D (Figure D.1). Table 4.1 gives is a brief description of the information that every table of the database holds.

Table Name	Description
User	Contains all the information about the user.
Deliverables	Represents the phase of the project that the user is in.
Technique	Contains the techniques that players can use to complete a deliverable.
User_has_ Technique	Stores the techniques that have already been chosen by the user and in which deliverable.
Deliverables_has_Technique	Contains the points of every technique in every deliverable.

Table 4.1: Initial model database tables description

4.2 Final Design

As previously mentioned, during the implementation process the requirements of the system changed because additional functionality was required. Because of the evaluation questionnaire, some features needed to be added and some needed to be removed or to change. (New and specific changes on the requirements can be found in Appendix C.) In particular, the game remained web-based, but instead of only helping to understand agile methods, it was also required to test the level of knowledge of the user before and after the completion of the Agile Game by the use of questionnaires to track their progress. In addition, the actual game users would have the opportunity to familiarise themselves with the most commonly used agile methods.

In particular, users would be able to choose one of the different agile methods and any of the techniques in every deliverable. Each technique is credited with a specific number of points depending on the phase of the project. At the end of the game, users are able to see their overall progress and the score of the top 10 players.

As the system was becoming more and more complex, in order to avoid and reduce coupling, the code had to be broken into smaller files. This helped to ensure that all the files and functions interact with each other in an optimum way. Table 4.2 gives a small description of the functionality of every PHP file and Figure 4.2 a system map, representing the way that files interact with each other.

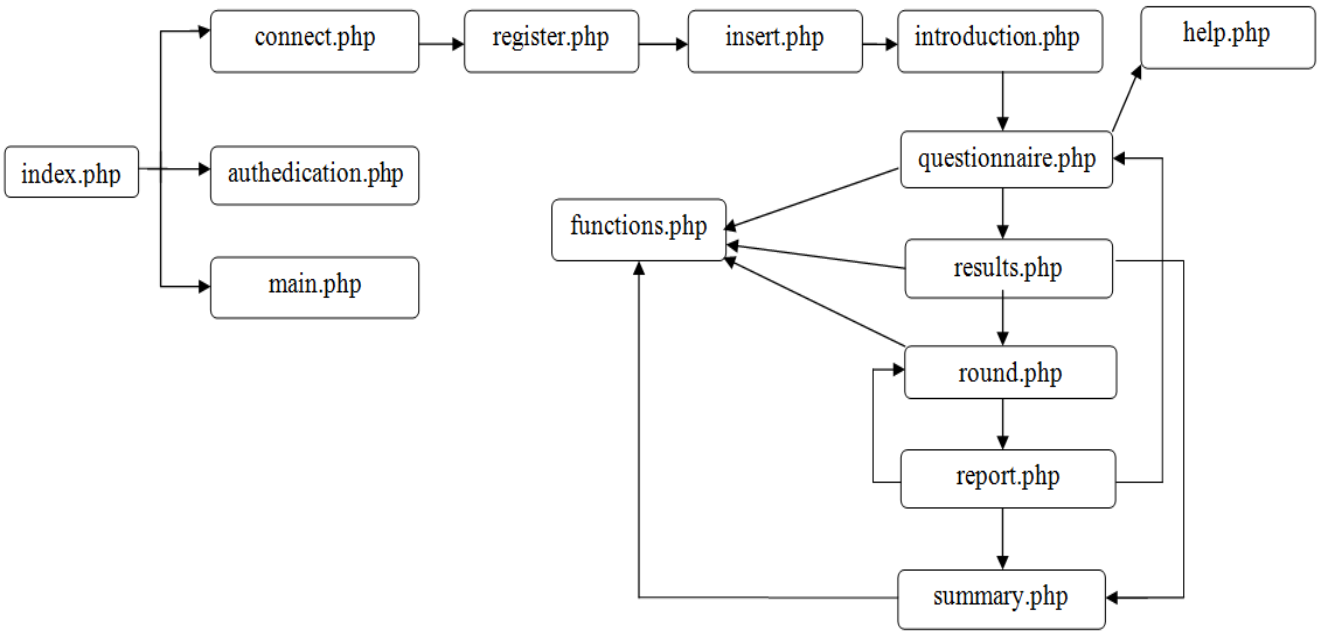

Figure 4.2: System map

File	Description
index.php	The main page of the website.
connect.php	Connects the website with the database.
introduction.php	Introduces the user to the Agile Game.
authentication.php	If user is already registered in the system, after inserting the correct username and password.
register.php	Registers the new user to the database.
insert.php	Checks if users are registered in the system after they inserted their username and if yes, they let them into the system.
functions.php	Contains all the necessary functions of the system.
help.php	Gives the user detailed information about agile methods.
questionnaire.php	Provides the questionnaire of the system.
results.php	Contains the results and the score of the user after the completion of the questionnaire.
main.php	Contains a summary of the progress of the user when they return to the system after logging out.
round.php	The main page of the game. The page where users select the methods and techniques.
report.php	Summarises the progress of the user in a deliverable.
summary.php	The final page of the game. Contains a summary of the overall progress of the user during the game.

Table 4.2: Description of PHP files

4.2.1 Final Database

The complete model of the database of the initial design can be found in Appendix D (Figure D.2) and the database schema on Appendix E. In the final model of the database, some more tables were added (Methods, User_has_Methods, Points, User_has_Points, Question, Answer and User_has_Score) to accommodate the required additional functionality. Below is a brief description of the information that every table of the database holds.

Table Name	Description
User	Contains all the information about the user.
Deliverables	Represents the phase of the project that the user is in.
Methods	Contains all the methods in every deliverable.
Technique	Contains the techniques that players can use to complete a deliverable.
User_has_Methods	Stores the methods that have already been chosen by the user and in which method and deliverable.
Points	Contains the points of every technique in every method in every deliverable.
User_has_Points	Stores the points that of every technique in every deliverable.
Question	Contains all the questions in the questionnaire.
Answer	Contains all the possible answers of the questionnaire.
User_has_Score	Stores the score of the user in the questionnaire.

Table 4.3: Final model database tables description

5 Implementation

5.1 Development Languages

For the implementation of the Agile Game, several development languages were used. Before deciding which programming languages were more appropriate for the implementation of the system, some research was conducted to investigate the advantages and disadvantages of each available development language.

At the beginning of the project, before defining the specific requirements and format of the system, it was assumed that the game would be implemented with Java SE, because the author had previous experience with it and was more familiar with its syntax and documentation, so valuable time would be gained. It would also allow the creation of a more interactive interface and insertion of more complex features in the game. However, as soon as the requirements were established, it was determined that the use of Java SE would restrict the users of the Agile Game because they do not have administrative rights to install programs on the School's machines, so they would not be able to access the game at any time. To avoid this problem, the game had to become a web-based application. In addition, there was the danger that the users' attention would be drawn by the Java interface rather than the actual functionality, quality and information of the game, losing its educational aim.

Because the Agile Game would be web-based, the most common scripting languages (PHP, ASP.NET& J2EE) were evaluated before deciding on the ideal one. The tables below represent the main advantages and disadvantages of these languages.

PHP	
Advantages	**Disadvantages**
Many libraries and frameworks	Few formal training courses
Recommended for small systems	Poor separation of rules
Open source	Low scalability
Cross platform	Limited handling of exceptions
Many books and online communities	
Support for objects and modularity	

Table 5.1: PHP advantages and disadvantages[3]

ASP.NET	
Advantages	**Disadvantages**
Good separation of roles	Complex model as it progresses
Visual development environment	Not recommended for small systems
Good support and training opportunities	Not recommended for non-Windows platforms
Many libraries and frameworks	
Scalability	

Table 5.2: ASP.NET advantages and disadvantages[4]

[3] Available from: https://secure.ecs.soton.ac.uk/notes/comp3018/

J2EE	
Advantages	**Disadvantages**
Good separation roles	Compatibility issues
Many libraries and frameworks	Not recommended for small businesses
Scalability	Proprietary
Good support and training opportunities	
Many books and online communities	
Open source and proprietary implementations	

Table 5.3: J2EE advantages and disadvantages[5]

It can be seen that all three languages have very strong advantages. The main difference between the three is that J2EE and ASP.NET are appropriate for complex and large systems, but the Agile Game is only a simple and small web application. These languages also provide very good separation of roles (between the designer and the coder), which in this system is not important since both the design and the code were implemented by the same person. ASP.NET provides a visual environment but for this system is of minor importance since it is a prototype and focuses more on the quality of the information that is provided and not on the interface. For these reasons, PHP was the most appropriate language for the implementation of the game.

Furthermore, a back-end database was needed to store all the necessary user and system information. A number of different kinds of database were considered, such MySQL, Microsoft Access and Oracle. From these technologies, the MySQL database was chosen because it is cross-platform, scalable, supports multiple user connections, the technology was known by the author, and it interacts effectively with the PHP scripting language. In addition, because the system is web-based, the use of HTML and CSS was essential to form the structure and the view of the website.

To make the system more interactive and appealing to the user, some client-side scripting languages were used, even though the system is a prototype. Thus, browser languages JavaScript and AJAX were used in order to validate forms and insert additional features in the system. It is important to note that the author had limited knowledge of the syntax and structure of PHP[6], JavaScript[7] and AJAX[8] languages and for this reason, some time was spent becoming familiar with them. The author consulted books that cover these subjects (Castro 2007; Nixon 2009; Welling 2009), online resources, as well as lecture notes[9].

[4] Available from: https://secure.ecs.soton.ac.uk/notes/comp3018/
[5] Available from: https://secure.ecs.soton.ac.uk/notes/comp3018/
[6] Available from: http://php.net/index.php
[7] Available from: http://www.w3schools.com/JS/default.asp
[8] Available from: http://www.w3schools.com/ajax/default.asp
[9] Available from: https://secure.ecs.soton.ac.uk/notes/comp3018/

5.2 Development Tools

Various development tools were used throughout the project. Others were used for the creation of the database, others for the implementation of the code and others for the writing of the report and documentation.

Specifically, at the beginning, for the implementation of the code Notepad2 was used, but as the implementation progressed and the code became more complex it was replaced by development using Dreamweaver. This program was chosen because it supports all the languages used by the system (PHP, HTML, CSS, JavaScript and AJAX) and every change in the code could be directly updated on the School's servers.

MySQL Workbench and MySQL Query Browser were used for the creation of the database tables and schema. The database is hosted by the School's Linuxproj server. In particular, Linuxproj includes an Apache web server that supports services like PHP and MySQL. The use of this server improves the security of the system because it is visible only within the ECS firewall and deals with storage instead of the administration of the system. Also, because the system is only visible within ECS, the system is accessible solely to students within the School. The database hosted by Linuxproj needs to be regularly backed up because all the data are deleted at the end of the academic year.

Microsoft Project was used for the project Gantt charts. The use of this tool was used with caution because of the wide functionality that it offers, so that the Gantt chart could become complex without focusing on the important project milestones. TortoiseSVN[10] online repository was used to store different versions of the project as it progressed as a back-up. For the creation and circulation of the evaluation questionnaire of the Agile Game, iSurvey[11] was used. This survey tool was created for the School of Psychology of the University of Southampton and can be used by all the members of the University. Its use ensures that all the participants are members of the University of Southampton.

Finally, the project brief, progress report, final project report and any other documentation of the project were created using Microsoft Word processor, Putty was used to access the database from Linuxproj, and WinSCP to access the public_html folder stored in the university filestore, while remote from ECS.

5.3 Feature Implementation

Implementation of the Agile Game resulted in the production of a considerable quantity of code. This section contains information about features that were considered the trickiest parts of the implementation and the ways by which they were resolved. All the files used to create the Agile Game were included on a CD ROM. An index to the CD contents can be found in Appendix L and additional screenshots of the system can be found on Appendix K. In addition, the online implementation of the system can currently be found at:

http://users.ecs.soton.ac.uk/ag2006/COMP3020/

[10] Available from: http://tortoisesvn.tigris.org/
[11] Available from: http://www.isurvey.soton.ac.uk/admin/

5.3.1 Username Availability

When new users try to register on the system, it checks if their username already exists in the database. If the username is available, players are notified that their details have been recorded so are able to start the game immediately. Initially, this process was implemented by redirecting the user from one page to another, resulting in a static outcome. Also, users had to first insert all their details and then get notified that the username was unavailable. To avoid that and make the system more interactive, some AJAX and JavaScript code was used to check the availability of the username in real-time while the user is typing (Figure 5.1).

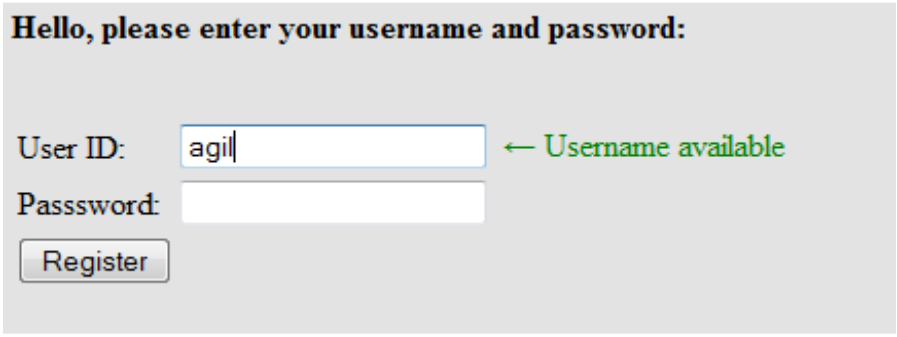

Figure 5.1: Username Availability

5.3.2 Questionnaire

The production of the questionnaire was very important for the project to fulfil its educational goal. For its construction, the system had to access the database multiple times to present the questions and the possible answers, thus reducing the efficiency of the system. Because the Agile Game is a small-scale project and the number of its expected users is limited, efficiency does not seem to be an issue. For the questionnaire, two nested **`for`** loops were used, one to access the database and print the question, and the other to print the available answers. Figure 5.2 represents the outcome and Figure 5.3 the code that was used. This method was used because it is simple and easy to read. If efficiency were an issue, then a joined query would be used to access the database the fewest possible times.

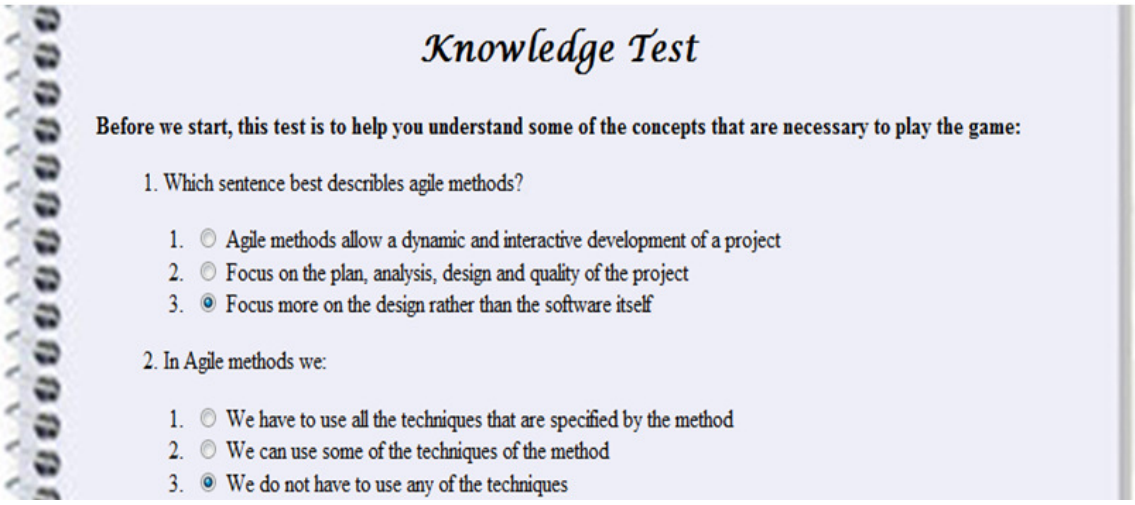

Figure 5.2: Questionnaire

```
//get number and description of the question where final = current_deliv
$query = "SELECT number, qtext  FROM `Question` WHERE final ='".$row1[0]. "'";
$result = mysql_query($query);
echo '<div class = "questions">';
//for every question get the possible answers
for ($i = 0; $i <mysql_num_rows($result);$i++){

$row = mysql_fetch_row($result); //number, qtext

//get all the possible answers of the question
$query2 = "SELECT otext FROM `Answer` WHERE question_numb ='".$row[0]. "' AND stage = '".$row1[0]."'";
$result2 = mysql_query($query2);

//start the number of the question from 1 instead of 0
$qno = $i+1;

//echo 'the question number'. $qno;

//print the number of question and question description
echo '  <p> '.$qno.'. '.$row[1].'</p>';
echo '<ol>';

//print all the possible answers of the question
for ($x=0; $x < mysql_num_rows($result2); $x++){

    $row2 = mysql_fetch_row($result2);
echo'
    <li><input type = "radio" name = "'.$row[0].'" value = "'.$row2[0].'" checked/>
    '.$row2[0].'</li>';
    }
    echo'</ol>';

    }

echo'<input type = "submit" id = "submit" value = "Check Results" >
    <input type="reset" name="reset" value="Clear" >
    </div>
      </form>';
```

Figure 5.3: Questionnaire code

5.3.3 Methods & Techniques

In the main page of the game, the user is required to select one of the available methods and any of its techniques. Because all the methods and techniques are available to the user, the amount of information displayed on the page was large and there was a possibility that a user could change their mind and select another method and its techniques, while having already selected one.

To avoid all these, it was decided that the techniques for each method would be hidden. Thus, only the methods would be visible to the user when they visit the main page (Figure 5.4). Next to every method, there is a radio button that if checked, reveals the techniques for the specific method (Figure 5.5 and Figure 5.6).

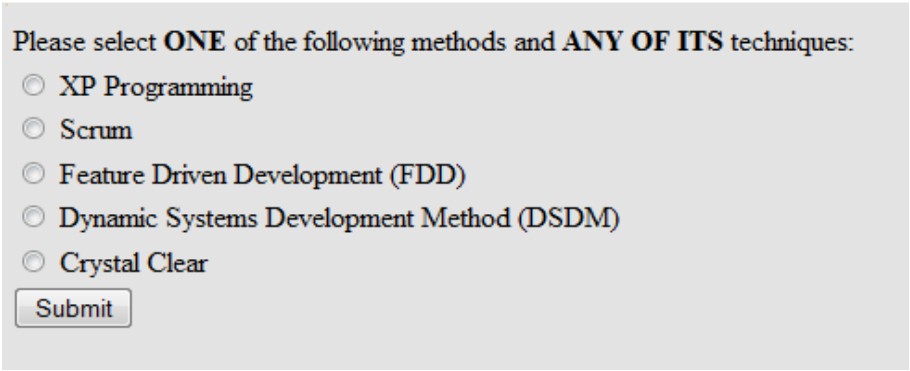

Figure 5.4: Methods

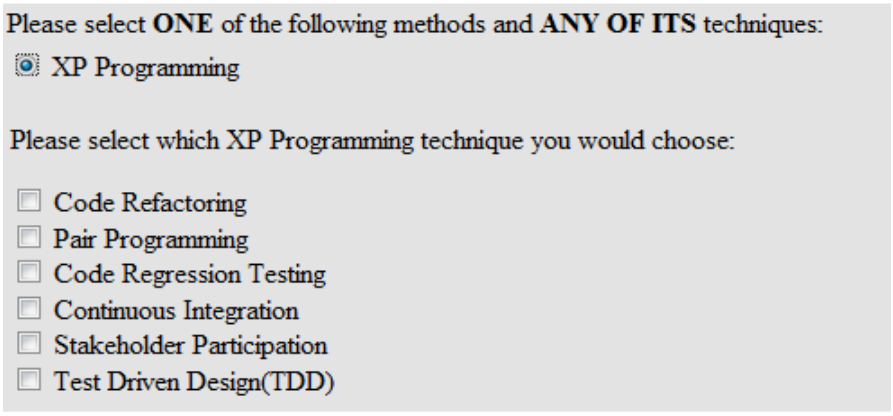

Figure 5.5: Methods & Techniques

```
$query = " SELECT meth_id FROM `Methods` WHERE delivid=".$row[0];
$result = mysql_query($query);

for ($i=0; $i< mysql_num_rows($result); $i++){
        $row = mysql_fetch_row($result);
        $methodid = $row[0];
        createSelections($methodid);

        }
function createSelections($methodid){

    $query = "SELECT method_id, tech_id, tech_desc, meth_desc FROM `Technique`
INNER JOIN  Methods ON meth_id=method_id WHERE method_id='".$methodid."'";
    $result = mysql_query($query);
    $row = mysql_fetch_row($result);


    echo'<tr><td><input type = "radio" name = "methods" value = "'.$row[0].'"
onfocus="show('.$row[0].')" ondblclick="uncheckRadio('.$row[0].')" /> '.$row[3].'
</td></tr>



    <tr id="a'.$row[0].'" style="display: none"><td><p>Please select which '.
$row[3].' technique you would choose:</p>';
    for ($i=0; $i< mysql_num_rows($result); $i++){
        echo '<input  type = "checkbox" name = "techniques[]" value = "'.$row[1]
.'"/> '.$row[2].'<br />';
        $row = mysql_fetch_row($result);
    }
    echo '</td></tr>';
}
```

Figure 5.6: Methods & Techniques code

The main difficulty was make the techniques disappear. After some research, it was determined that in PHP the radio buttons could not be unchecked and for this reason, some JavaScript was used. Also, in every phase of the project, each method and technique has a score. The score depends on the chosen method and the deliverable that the user is implementing, stored in the table Points (Appendix D.2). This table contains the points that correspond to every technique in every method and in every deliverable. Figure 5.7 shows the contents of this table and Figure 5.8 shows how the points are calculated.

```
mysql> select * from Points;
+----------+----------+---------+---------+--------+
| point_id | deliv_no | meth_no | tech_no | points |
+----------+----------+---------+---------+--------+
|        1 |        1 |       1 |       1 |      0 |
|        2 |        1 |       1 |       2 |      0 |
|        3 |        1 |       1 |       3 |      0 |
|        4 |        1 |       1 |       4 |      0 |
|        5 |        1 |       1 |       5 |     25 |
|        6 |        1 |       1 |       6 |      0 |
|        7 |        2 |       2 |       7 |      5 |
|        8 |        2 |       2 |       8 |      9 |
|        9 |        2 |       2 |       9 |      0 |
|       10 |        2 |       2 |      10 |      0 |
|       11 |        2 |       2 |      11 |      8 |
|       12 |        2 |       2 |      12 |      3 |
|       13 |        3 |       3 |      13 |      4 |
|       14 |        3 |       3 |      14 |      8 |
|       15 |        3 |       3 |      15 |      4 |
```

Figure 5.7: Table Points

```
            //select point_id
            $query = "SELECT point_id, points FROM `Points` WHERE  deliv_no = '".
$currentdel."' AND meth_no = '" .$method."' AND  tech_no = '" .$technique[$i]."'";
            $result = mysql_query($query);
            $rows1 = mysql_fetch_row($result);


            $score = $score + $rows1[1];


                //record user points
                $query = "INSERT INTO `User_has_Points` VALUES ( '" .$user. "', '" .
$rows1[0]."')"; //name, point_no
                $result = mysql_query($query);

                }
                echo $score;

                //calculate the points of each player
                $query = "SELECT sum(points) FROM  User_has_Points INNER JOIN Points
 ON point_no = point_id WHERE name = '" .$user. "' ";

                $result = mysql_query($query);
                $row = mysql_fetch_row($result);

                echo '<p><b>The points that you have gained overall until this point
 are:</b></p>  ' ;
                echo $row[0];
    }
```

Figure 5.8: Points calculation

6 Testing

Testing is a very important phase of the lifecycle. In XP methodology, tests are created before writing any code; as soon as some functionality of the system is implemented, it is tested. As Kent Beck says "We will write tests before we code, minute by minute. We will preserve these tests forever, and run them all together frequently. We will also derive tests from the customer's perspective" (Beck 2000).

6.1 Testing Methodologies

For this system, multiple types of testing methodology were used to thoroughly test the system. In particular, black box testing was used to test the system functionality. In black box testing, the focus is on the outputs of the system and not its internal functionality (Test cases F.1 – F.47). In addition, white box testing and boundary analysis were used to test the upper boundaries of the database values (Test cases F.48 – F.52). Finally, specification testing was used to check the initial functional (Test cases F.53 – F.78) and non-functional requirements (Test cases F.79 – F.85) of the system against the final outcome and ensure that the system fulfils the functionality as originally specified.

6.2 Summary of Test Cases

A full description of the various test cases can be found on Appendix F, but below there is a summary of them.

Test Type	No	Total No of test cases	Passed	Failed
System Testing	F.1 – F.47	47	46	1
Database Testing	F.48 – F.52	5	5	0
Functional Requirements Testing	F.53 – F.78	25	25	0
Non-Functional Requirements Testing	F.79 – F.85	6	5	1
Total:		**83**	**81**	**2**

Table 6.1: Summary of Test Cases

7 Evaluation

The evaluation of the overall system and progress is one of the most important aspects of this project. To be more specific, evaluating the final outcome of the project is essential to assess how successful the project has been. To investigate this, two different methods have been used:

- Evaluation questionnaire
- Interviews

7.1 Overview

One can say that the project has been successful because all the deadlines have been met, despite some minor changes in the initial plan. In addition, the system is complete and functional to 98% of its requirements, as the test cases showed (see Section 6.2). However, these metrics do not constitute a very precise way to evaluate user satisfaction. For this reason, the use of an evaluation questionnaire was necessary. In particular, as soon as the system was complete, a questionnaire was given to some students to evaluate the game. The questionnaire can be found in Appendix G and its detailed results are given in Appendix H. The questions concerned the system's interface, its usability and whether it managed to achieve its educational aim.

The results of the questionnaire indicated that changes concerning the amount of feedback that a player gets about their decisions during the game were essential but despite the criticisms the game got positive feedback and it was rated quite high overall (Figure H.20). Also, there were comments on the game's interface (e.g. "it is very simple") which are important, but the game's interface was not its primary goal. Its aim was to present helpful and meaningful information to familiarise students with agile methods. For this reason, following the evaluation questionnaire, some improvements were made to the database and the code to meet the user requirements.

After these changes, the author wanted to conduct another evaluation, but this was not possible due to time constraints. As a substitute, the author interviewed a small number of people for their opinion on the additional changes, using the same evaluation questionnaire as before (Appendix G). The users again drew attention to the user interface, but the comments concerning the quality and the amount of information that the system offers were positive, although some still believed that a considerable amount of prior knowledge was assumed. Appendix H provides a more a more detailed evaluation.

7.2 Questionnaire & Interview Results

Appendix H contains all the detailed information from the results of the evaluation questionnaire. In this section, only a summary of the results is illustrated. Only 17 users completed the questionnaire. Ideally, this type of evaluation would be made by a large number of users and over a long period of time, but this was not possible due to time constraints.

Specifically, it was noticed that players were more familiar with traditional project management methods rather than with agile methods, even though they have been taught about the latter whilst at university (Figures 7.1 and 7.2).

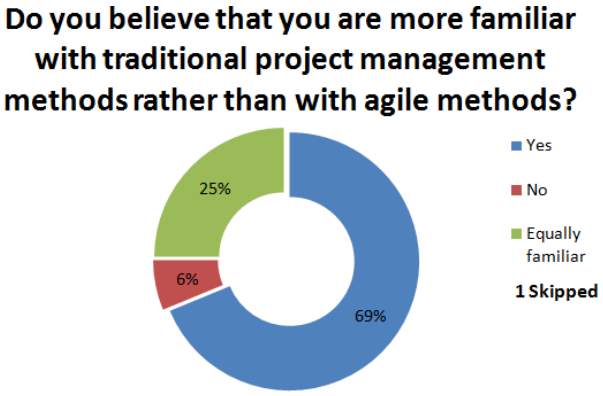

Figure 7.1: Questionnaire Results

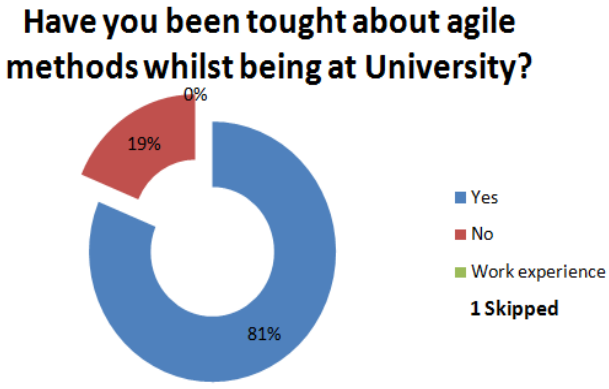

Figure 7.2: Questionnaire Results

The majority were familiar with the XP method (which is taught at the University of Southampton as part of the agile methods syllabus) and some of its techniques, which are also used in traditional project management (Figures 7.3 & 7.4).

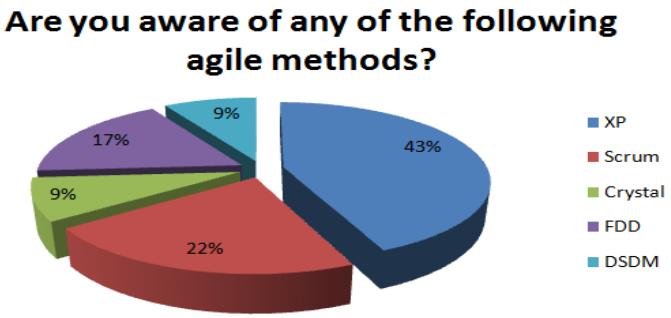

Figure 7.3: Questionnaire Results

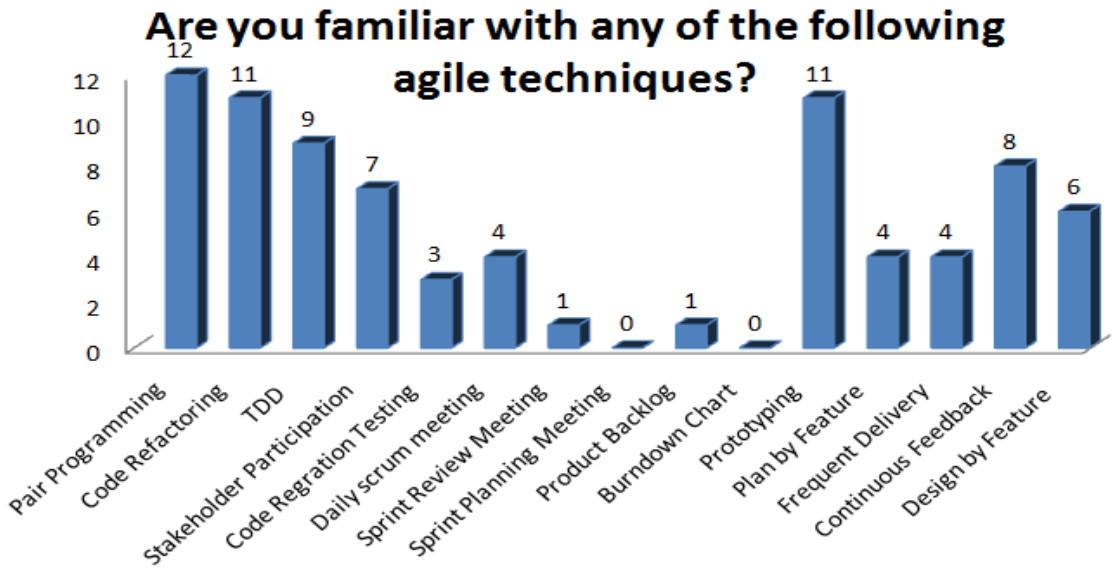

Figure 7.4: Questionnaire Results

While playing the game, 67% of the users found that they could easily navigate through the website (Figure 7.5), and more than half of them believe that the interface was pleasant (Figure 7.6).

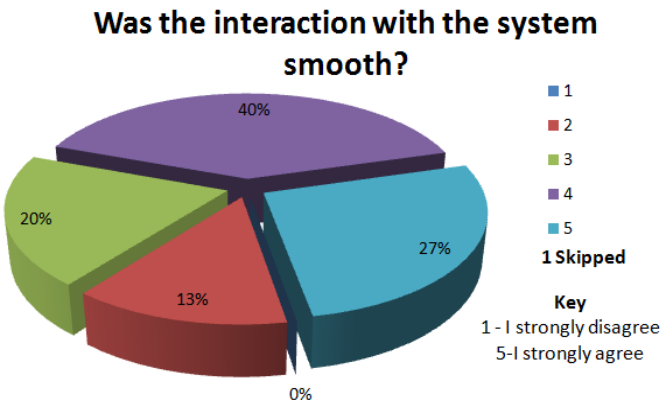

Figure 7.5: Questionnaire Results

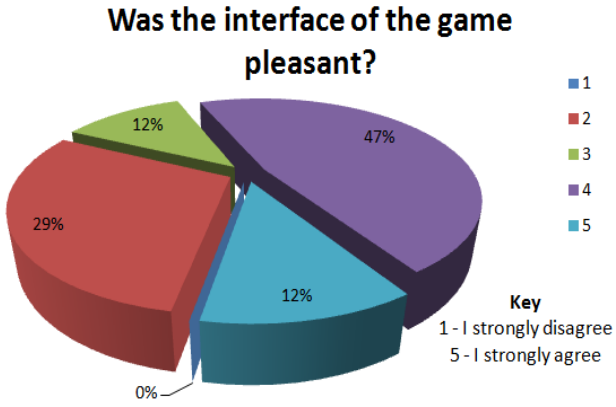

Figure 7.6: Questionnaire Results

As previously mentioned, the majority felt that too much prior knowledge was assumed (Figure 7.7) and only 14% strongly agrees that the provided help resources were useful (Figure 7.8) and there was satisfying amount of feedback in every step of the game.

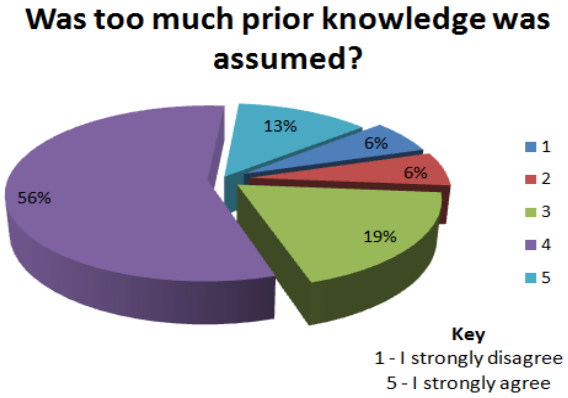

Figure 7.7: Questionnaire Results

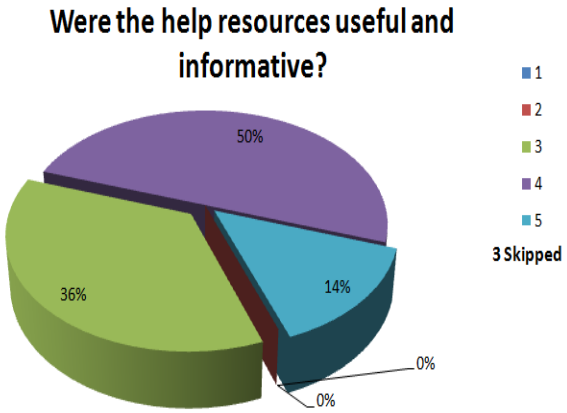

Figure 7.8: Questionnaire Results

At the end of the game, users were almost equally more confident with all of the agile methods (Figure 7.9) and most of their techniques (Figure 7.10), while 44% believe that it fulfils its initial aim (Figure 7.11) and the system was rated at about 7 out of 10 (Figure 7.12).

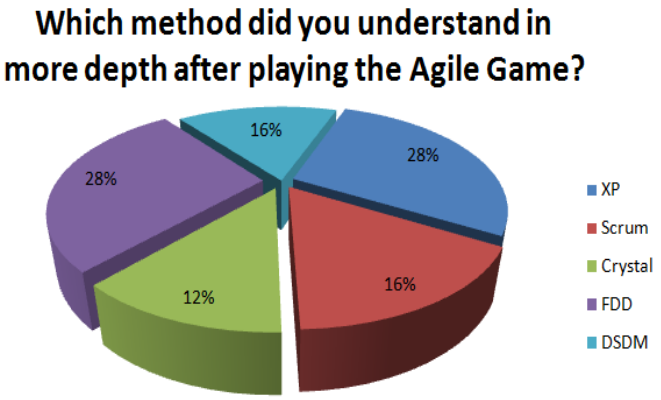

Figure 7.9: Questionnaire **Results**

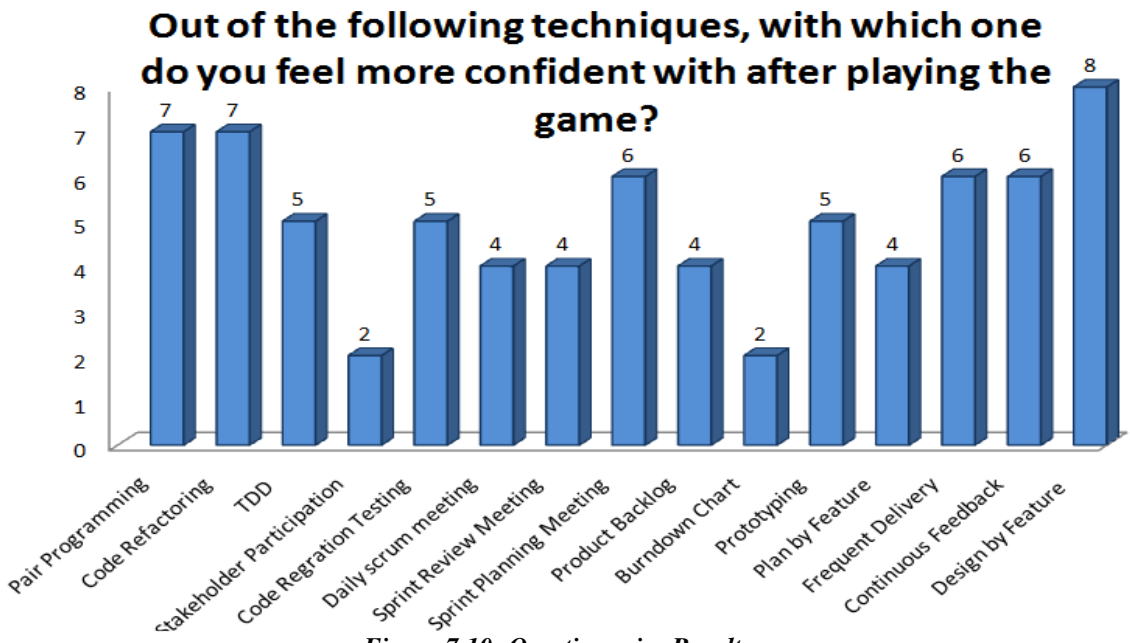

Figure 7.10: Questionnaire Results

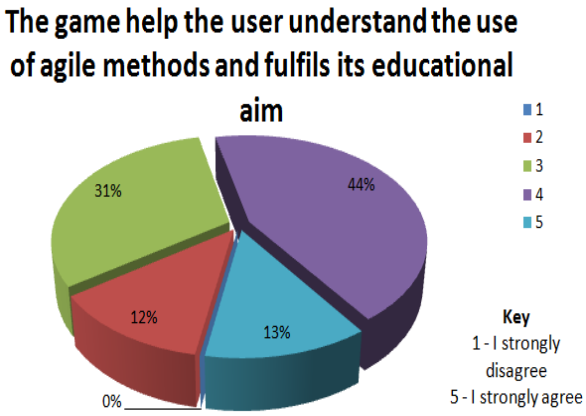

Figure 7.11: Questionnaire Results

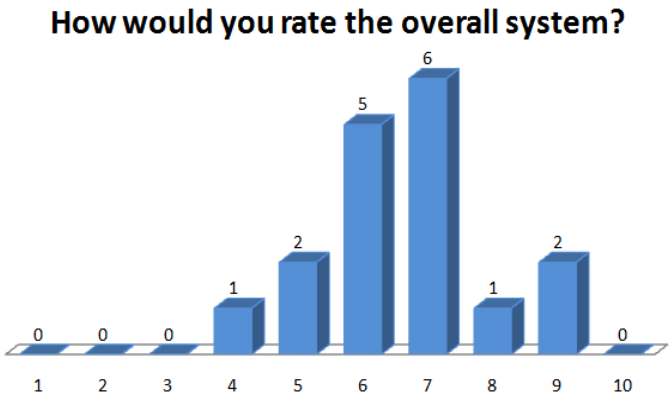

Figure 7.12: Questionnaire Results

Furthermore, in the additional interview only 4 students took part due to time constraints (for the interview the evaluation questionnaire in Appendix G was used

again due to time constraints). Because the sample was small, it was considered that including graphs of their opinions would not represent accurate results. The interviewees found that, after the changes, they could find feedback that was more accurate at every step of the game and an explanation of how their score was formed. In addition, they believed that the information in the help resources was more comprehensive. However, there were still comments about the game's interface stating that it could be improved, but due to time constraints the author focused more on the quality of the information of the game rather than on its interface (Appendix H).

7.3 Project Management

Overall, the project was successful in terms of project management. Even though there were some differences in the final plan compared to the initial plan, they did not affect the overall schedule so all the tasks were delivered on time (More detailed explanation can be found in Appendix B).

7.4 Project Goals

The evaluation of the project shows that the project was successful. This can also be seen by the table below that illustrates that all the initial system goals (Appendix C) have been met:

Project Goal	Status	Justification
Meet all the deadlines of the project.	Met	As can be seen from Figure B.6, all deadlines were met.
Implement a project following the planned design.	Met	The Agile Game was implemented following the specified requirements.
Meet all the system goals.	Met	As can be seen from Figure C.3, the project has met all its initial system goals.
Meet all the project goals.	Met	As can be seen from Figure C.4, the project has met all its initial project goals.
Provide a fully functional prototype of the Agile Game.	Met	The summary of the test cases of the Agile Game illustrates that only 2% of the tests failed.
Create a game that helps students understand agile methods in more depth.	Met	Almost 60% of the users agree that the game helps to understand agile methods.

Table 7.1: Achieved Project Goals

8 Conclusion & Future Work

The evaluation showed that the project was a success in many aspects. The aim of the project was achieved by creating a prototype of a system that simulates the use and the impact of agile methods and their techniques within the project lifecycle. The planning, design, implementation, testing and evaluation of the project satisfied the requirements that were set. Even though in the first evaluation users were not very satisfied with the information on agile methods provided, after reviewing the system, their opinion changed and their comments were positive. Almost 98% of the tests were successful and all of the project and the system goals were met. Thus, from these and from the user reviews it was demonstrated that the Agile Game managed to familiarise its users with agile methods.

In this project, the author managed to research a broad part of project management methods, both traditional and agile, other educational games and the role of games in education. These helped the author understand and specify which features were important and which had to be incorporated in the Agile Game. In addition, the author had the opportunity to test and practice project management skills across the duration of the project, to manage and keep on schedule and meet the deadlines producing a good quality system that satisfies its requirements and goals. Finally, through this project, the author had the opportunity to learn new scripting languages PHP, JavaScript and AJAX, where there was no previous experience and managed to enrich existing knowledge of HTML, CSS and MySQL.

The Agile Game is a promising system/game and has many aspects that can be improved. In particular, specific information on which method and techniques are more appropriate to use on each phase of a project, or which method is more suitable for a specific type of project, could be provided by enterprises leading to more detailed and accurate scores for every user choice during the game. Because it is web-based, the Agile Game could become more interactive, implementing it with 3D graphics or with other technologies such as Flash. Finally, the system could be improved by accommodating a multiplayer game for a team of players, and as soon as a project has been successfully completed, the team could proceed to the implementation of a new, more competitive project.

References

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002) *Agile Software Development Methods* Finland: VTT Technical Research Centre of Finland Available from: http://www.pss-europe.com/P478.pdf [Accessed: March 2010].

Aguanno, K., (2004) *Managing Agile Projects* USA: Multi-Media Publications Inc.

Ambler, S.W. (2008) *Agile Adoption Rate Survey Results: February 2008* Available from: http://www.ambysoft.com/surveys/agileFebruary2008.html [Accessed: March 2010].

Ambler, S.W. (2007) *Agile Adoption Rate Survey Results: March 2007* Available from: http://www.ambysoft.com/surveys/agileMarch2007.html [Accessed: March 2010].

Ambler, S.W. (2005-2009) *Feature Driven Development (FDD) and Agile Modelling* Canada: Ambysoft Inc. Available from: http://www.agilemodeling.com/essays/fdd.htm [Accessed: October 2009].

Ambler, S.W. (2006) *Agile Adoption Rate Survey Results: March 2006* Available from: http://www.ambysoft.com/surveys/agileMarch2006.html [Accessed: November 2009].

Barnes, T., Powell, E., Chaffin, A. & Lipford, H. (2008) Game2Learn: improving the motivation of CS1 students *GDCSE '08: Proceedings of the 3rd international conference on Game development in computer science education* ACM.

Basturk, R. (2005). *The Effectiveness of Computer-Assisted Instruction in Teaching Introductory Statistics*. Educational Technology & Society, 8 (2), 170-178.

Beck, K. & Andres, C. (2004) *Extreme Programming Explained.* 2nd ed. U.S.: John Wait, Pearson Education.

Beck, K. (2003) *Test – Driven Development: by example* Boston: Addison- Wesley.

Beck, K. (2000) *Extreme Programming Explained: Embrace Change*. Boston: Addison-Wesley.

Becker, K. (2001) Teaching with Games: the Minesweeper and Asteroids Experience *Journal of Computing Sciences in Colleges* Volume 17 Issue 2

Castro, E. (2007) *HTML, XHTML & CSS* 3rd ed. USA: Peachpit Press.

Chang, M. (2010) Agile and Crystal Clear with Library IT Innovations In *Proceedings of VALA2010 Conference* [online] Available from: http://www.vala.org.au/vala2010/papers2010/VALA2010_14_Chang_Final.pdf [Accessed: April 2010].

Clua, E., Feijó, B., Rocca, J., Schwartz, J., Perlin, K., Tori, R. & Barnes, T. (2006) Game and Interactivity in Computer Science Education IN: *SIGGRAPH '06: SIGGRAPH 2006 Educators program*. ACM.

Cohn, M. (2007 a) *Agile Estimating and Planning* Upper Saddle River: Pearson Education, Inc.

Cohn, M. (2007 b) *User Stories Applied* Boston: Pearson Education, Inc.

Cockburn, A. (2009) *Crystal Methodologies* Available from: http://alistair.cockburn.us/Crystal+methodologies [Accessed: October 2009].

Cockburn, A. (2002) *Agile Software Development* Boston: Addison-Wesley.

Dantras, A., Barros, M. & Werner, C. (2004) *A Simulation-Based Game for Project Management Experiential Learning* [online] Available from: https://reuse.cos.ufrj.br/prometeus/publicacoes/SEKE04-AlexDantas.pdf [Accessed: March 2010].

Gotterbarn, D., (2008) Thinking professionally: a real problem with video games; not murder, not torture... *SIGCSE Bulletin* Volume 40 Issue 4 ACM.

Fowler, M. Beck, K., Brant, J., Opdyke, W. & Roberts, D. (2004) *Refactoring: Improving the Design of Existing Code* Boston: Addison-Wesley.

Fowler, M. [n.d.] *Refactoring Home Page* [online] Available from: http://www.refactoring.com/ [Accessed: March 2010].

Galvão, J. R., Martins, P. G. & Gomes M. R. (2000) Modelling Reality with Simulation Games for a Cooperative Learning *WSC '00: Proceedings of the 32nd conference on Winter simulation* ACM.

Garratt, P. W. (1995) *Simulation & Gaming - The Software Management Game*, Sage Publications.

Henson, V. L. (2008) *Agile Bad* Available from: http://www.agiledad.com/Documents/AgileDadAgileMethod.pdf [Accessed: March 2010].

Highsmith, J. (2002) *Agile Software Development Ecosystems* Indianapolis: Addison Wesley.

Jeffries, R. (2001) *XProgramming.com: An Agile Software Development Resource* [online] Available from: http://xprogramming.com/xpmag/whatisxp [Accessed: March 2010].

Lewis, J. P. (2007) *Fundamentals of Project Management.* 3rd ed. New York: AMACOM.

Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F., Tesoriero, R., Williams, L. & Zelkowitz, M. (2002) Empirical Findings in Agile Methods *Computer Science Notes* vol. 2418/2002, Springer, Berlin pp.81-92.

MacDonald, K. J. ([n.d.] a) *Agile Method Brief – Extreme Programming* [online] Available from: http://www.projectconnections.com/templates/detail/agile-extreme-programming.html [Accessed: April 2010].

MacDonald, K. J. ([n.d.] b) *Agile Method Brief-Feature Driven Development* [online] Available from: http://www.projectconnections.com/templates/detail/agile-techniques-fdd.html [Accessed: March 2010].

Martin, A. (2000) A simulation engine for custom project management education *International Journal of Project Management 18 pp.201-213* [online] Available from: http://www.sciencedirect.com/science/article/B6V9V-3YMW43B-7/2/561cfd9d3dc8f2ed4b27391de1305f34 [Accessed: April 2010].

McFarland, D. S. (2008) *JavaScript the missing manual* CA: Pogue Press O'Reilly.

MSDN (2010 a) *Regression Testing* Available from: http://msdn.microsoft.com/en-us/library/aa292167%28VS.71%29.aspx [Accessed: March 2010].

MSDN (2010 b) *Continuous Integration using Team Foundation Build* Available from: http://msdn.microsoft.com/en-us/library/ms364045%28VS.80%29.aspx [Accessed: March 2010].

Nixon, R. (2009) *Learning PHP, MySQL and JavaScript* CA: Pogue Press O'Reilly.

Oblinger, D. (2004). The Next Generation of Educational Engagement. *Journal of Interactive Media in Education* (8) Available from: http://jime.open.ac.uk/2004/8/oblinger-2004-8-disc-paper.html [Accessed: March 2010].

Schwaber, K. (2004) *Agile Project Management with Scrum* USA: Microsoft Press.

Scrum Alliance (2009) *Scrum Alliance: transforming the world to work* [online] Available from: http://www.scrumalliance.org/pages/what_is_scrum [Accessed: April 2010].

Serena (2007) *An Introduction to Agile Software Development* [online] Available from: http://www.serena.com/docs/repository/solutions/intro-to-agile-devel.pdf [Accessed: 26th October 2009].

Shine Technologies (2004) *Agile Methodologies Survey Results* Available from: http://www.agilealliance.org/system/article/file/1121/file.pdf [Accessed: March 2010].

Somerville, I. (2007) *Software Engineering* 8th ed. Essex, England: Pearson Education Limited.

Somerville, I. (2001) *Software Engineering* 6th ed. Essex, England: Pearson Education Limited.

Squire, K., (2003) *Video Games in Education* International Journal of Intelligent Simulations and Gaming (2) 1 Available from: http://website.education.wisc.edu/kdsquire/research.html [Accessed: April 2010].

Parsons, D. & Cranshaw, M. (2008) *Agile Technique Hour* Available from: http://www.massey.ac.nz/~dpparson/agilehour.htm [Accessed: November 2009].

Parsons, D., Ryu, H. & Lal, R. (2007) *The Impact of Methods and Techniques on Outcomes from Agile Software Development Projects* [online] Available from: http://www.massey.ac.nz/~dpparson/ParsonsRyu.pdf [Accessed: November 2009].

Peeters, V. & Cauwenberghe, P. V. (2008) *The XP Game* Available from: http://www.xp.be/xpgame.html [Accessed: November 2009].

VersionOne (2009) *State of Agile Survey 2009* Available from: http://trailridgeconsulting.com/surveys/state-of-agile-development-survey-2009.pdf [Accessed: March 2010].

Virvou, M, Katsionis, G. & Manos, K. (2005) Combining Software Games with Education: Evaluation of its Educational Effectiveness *Educational Technology & Society* 8 (2) pp.54-65 Available from: http://www.ifets.info/journals/8_2/5.pdf [Accessed: April 2010].

Voigt, B. J. J., (2004) *Dynamic System Development Method* [online] Available from: http://www.ifi.uzh.ch/req/courses/seminar_ws03/14_Voigt_DSMD_Ausarbeitung.pdf [Accessed: March 2010].

Wake, W. C. (2001) *Extreme Programming Explored* Upper Saddle River: Pearson Education Corporate.

Wells, D., (1997-1999) *Acceptance Tests* Available from: http://www.extremeprogramming.org/rules/functionaltests.html [Accessed: November 2009].

Welling, L. & Thomson, L. (2009) *PHP and MySQL Web Development* USA: Pearson Educational Inc.

Williams, L., Kessler, R. R., Cunningham, W. & Jeffries, R. (2000) *Strengthening the Case for Pair Programming* IEEE Software, vol. 17, no. 4, pp. 19-25 Available from : http://www.cs.utah.edu/~lwilliam/Papers/ieeeSoftware.PDF [Accessed: March 2010].

Appendix A – Background Research and Literature Search

A.1 Traditional Project Management Models

A.1.1 Waterfall Model

As mentioned in section 2.2, the Waterfall model is a characteristic example of a traditional project management model (Figure 2.1). This model represents the fundamental processes of a specification, development, validation and evolution and represents them as phases of a project, i.e. requirements, design, implementation, verification and maintenance (Somerville 2001).

The Waterfall model offers a complete analysis of the user's requirements. Even though this analysis is usually time-consuming, it offers "well-documented information" that can be used in the design of the project. This way, the program developers have a very precise design that can be used in the implementation and testing phases of the project. However, it is common that customers change their minds during the implementation of the project, because either the market needs have changed while the project is progressing, or they were not sure about what kind of system they needed exactly. For this reason, the requirements change and the Waterfall model cannot adapt to the new changes because the requirements have been determined at the beginning of the project and the delivery is one fully functional system (Aguanno 2004).

In contrast, agile methods are more flexible to changes because their development phase can be broken into many small sections, where each section delivers a fully functional part of the final deliverable. This incremental and iterative development allows alteration of the requirements, reducing their impact. As Aguanno says, agile methods are like "taking the processes behind Waterfall and repeating it throughout the development process (Aguanno 2004).

A.1.2 Spiral Model

Spiral is another traditional project management model. Unlike Waterfall, Spiral does not define all the requirements and the entire system at the beginning of the project but instead it first prioritises all the features of the system by risk (while agile methods prioritise feature by importance and functionality) and focuses on documentation. Also, this model includes long phases of requirements specification and design (Aguanno 2004).

Figure A.1 shows a typical Spiral lifecycle:

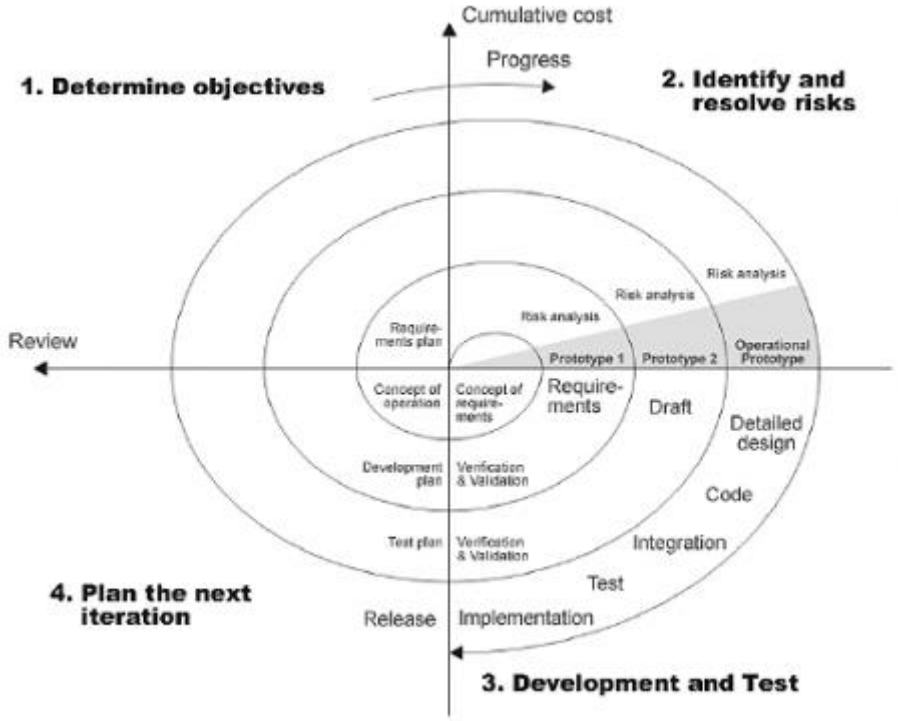

Figure A.1: Spiral Model[12]

The model has four phases (Requirements, Risk analysis, Development/Testing and Evaluation) which a project passes through repeatedly. In particular, at the beginning of the project, the requirements of the system are defined and a design is constructed based on them. The project constraints like budget, quality, deadlines, are defined as well as all the risks that might occur during the project development. Further, the risks are prioritised and if there are any significant project risks, a prototype might be constructed to identify and resolve the sources of the risk. Then, the software is implemented and tested and finally evaluated by the user.

A.1.3 V Model

This model is characterised by its sequential execution of deliverables. In particular, every phase of the project needs to be finished, completed and tested before proceeding to the next one. It is easy to use because it has specific deliverables, but it is not flexible to changes and it is difficult and expensive to alter the requirements of the system as soon as the implementation has started.

In addition to Waterfall model, in the V Model the attention is drawn to testing which takes place in every phase from the beginning of the lifecycle until the development of the software (Figure A.2). As soon as all the requirements and the design phases are complete (System Design, Architecture Design and Module Design), the software is implemented. When the implementation is complete, the software is validated against the tests that have already been created in the verification phase.

[12] Available from: http://accuracyandaesthetics.com/wp-content/uploads/2008/05/spiral_model_boehm_1988.jpg

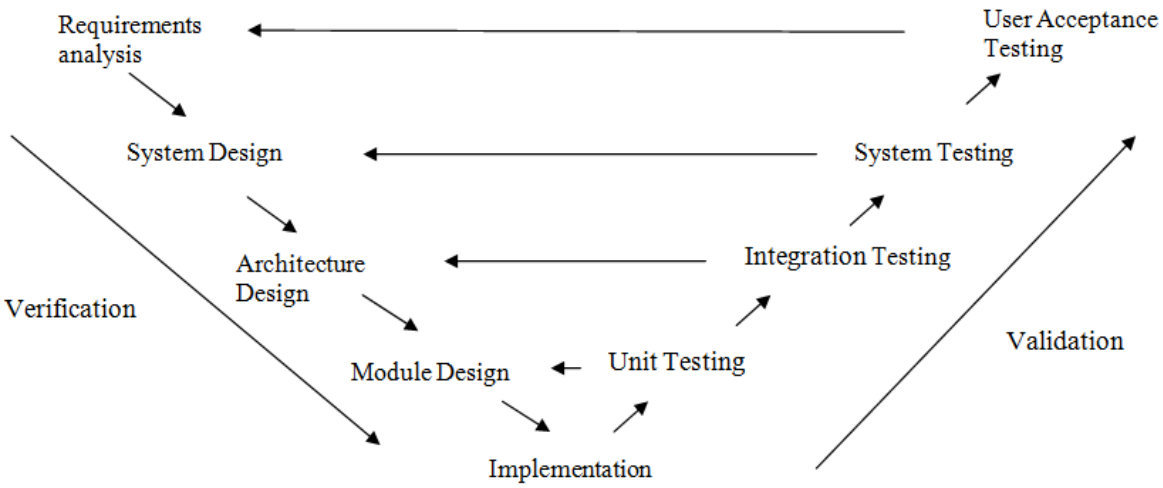

Figure A.2: V Model

A.2 Agile Methods

Agile methods are known for their ability adapt to any changes that might occur, because of their incremental delivery and small releases of the product during the development phase. This way, the impact of the change can be small, resulting to saving time and cost. With agile methods, the product is developed in multiple lifecycles, called iterations. Each iteration builds on the previous one to produce the final product, so the development becomes iterative and incremental (Aguanno 2004).

In addition, stakeholder participation is very important in agile methods. Every iteration is very short, in order to get feedback from the customer about the product, to ensure that the product is developed satisfying the user requirements, and to reduce the risk and impact of a change in the requirements (Aguanno 2004).

A.2.1 Extreme Programming (XP)

The basic features of XP have been described in section 2.3.1. Here, the different XP techniques will be examined. In particular:

- **Code Refactoring:** Code refactoring is the change to the code of an existing software system, without changing its external functionality. “The essence is improving the design of the code after it has been written” (Fowler et al. 2004; Fowler [n.d.]). It also improves maintainability, extensibility and regular testing before new code is integrated into the system (Wake 2001), which helps to minimise the number of faults in the system.

- **Pair Programming:** Pair programming is “when two programmers work side by side at one computer, continuously collaborating on the same design, algorithm, code or test” (Williams et al. 2000). Working in pairs helps both to understand and improve all the code as needed (Jeffries 2001). Surveys showed that implementing in pairs improves the productivity and the quality of the product (Williams et al. 2000).

- **Code Regression Testing:** is a procedure that needs to take place every time that some part of the code is altered. This can be done by running already existing tests with the modified code, to identify if any features that previously were working do not pass the tests and by writing new tests where necessary (MSDN 2010 a).

- **Continuous Integration:** is the software engineering practice that helps the quick delivery of the product. It is done when developers integrate their code and add more functionality and deliver a functional part of the project (MSDN 2010 b).

- **Stakeholder Participation:** is when the clients/stakeholders cooperate with the development team to define the requirements of the system to be implemented and give feedback during the development phase.

- **Test Driven Design (TDD):** The motto of this technique is "Red, Green, Refactor" where red means to create a test that fails, green is to implement some code and make the test pass, and refactor means to change the code to remove any duplication and improve its design, ensuring that all the tests still pass. From the previous, one can understand that TDD is the use of automated unit tests that help reduce coupling and verify that at any point the code is fully functional (Beck 2003).

A.3 User Stories vs. Use Cases

Some research was conducted to decide whether to use user stories or Use cases for the specification of the requirements. It was decided that the requirements should be in the form of user stories because they are mostly employed on projects developed with agile methods. They also represent the non-functional requirements of the system compared with the Use cases that cover only the functional requirements. Furthermore, user stories are normally written by the customers but here this is not applicable. They are written in a simple way so both customers and designers can understand them. They also include an estimate of the effort for a task, and customers prioritise them (Cohn 2007 b).

A.4 Agile Statistics

Many surveys have been conducted to identify the level of adoption of agile methods, their effectiveness, and the way that their development affects the project compared to traditional management methods. In 2008, in a survey conducted by Ambler it was shown that 68.5% of the responders are currently using agile methods in their teams (similar figures were shown in the same survey in 2007) and that almost 80% of these projects were successful (Ambler 2008; Ambler 2007). The most common reasons that agile projects fail, was found to be either because there is lack of expertise of the method that is being used, or because the company did not adopt all the principles of that particular method (VersionOne 2009). More than half of the responders (60%) state that the productivity of their team was a bit higher than traditional management methods, and 22% claim that the productivity is much higher (Ambler 2008). In addition, a bit more than 80% of responders stated that business satisfaction was slightly

or significantly higher with the use of agile methods, while only 1% felt that the adoption had negative effect (Shine Technologies 2004).

As for the quality of the final deliverable, slightly less than half state that the quality of the product was somewhat higher, and 29% believe that it was much higher when only 10% state that the quality is much lower (Ambler 2008). The previous figures contrast with the claim by some people that agile methods are used on projects where the final quality is not an issue or on projects with low quality. Finally, 40% state that the overall cost of the project did result in any change by the adoption of agile methods and a considerable 32% found that the overall cost was somewhat lower (Ambler 2008).

Appendix B – Management of this Project

In this section are the detailed plans of the project. Figure B.1 shows the Gantt chart that was created at the very beginning of the project. It represents a draft estimation of planning. Figure B.2 shows the initial plan for Semester One, and Figure B.3 how plans changed during the semester. Figure B.4 shows the plan of the remainder of the project in Semester Two, Figure B.5 represents the changes on the plan at the end of semester Two and Figure B.6 shows how the plan was actually developed.

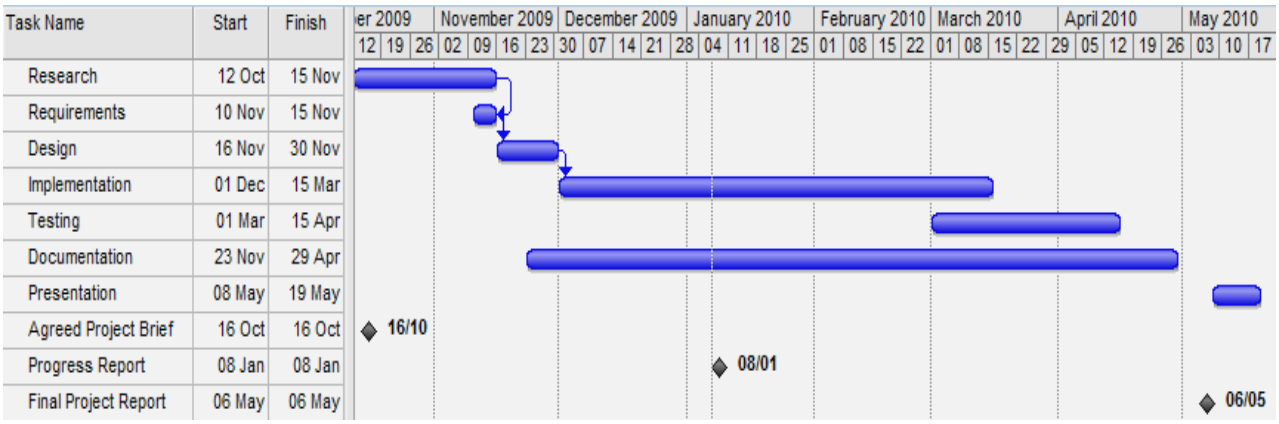

Figure B.1: Draft Gantt chart

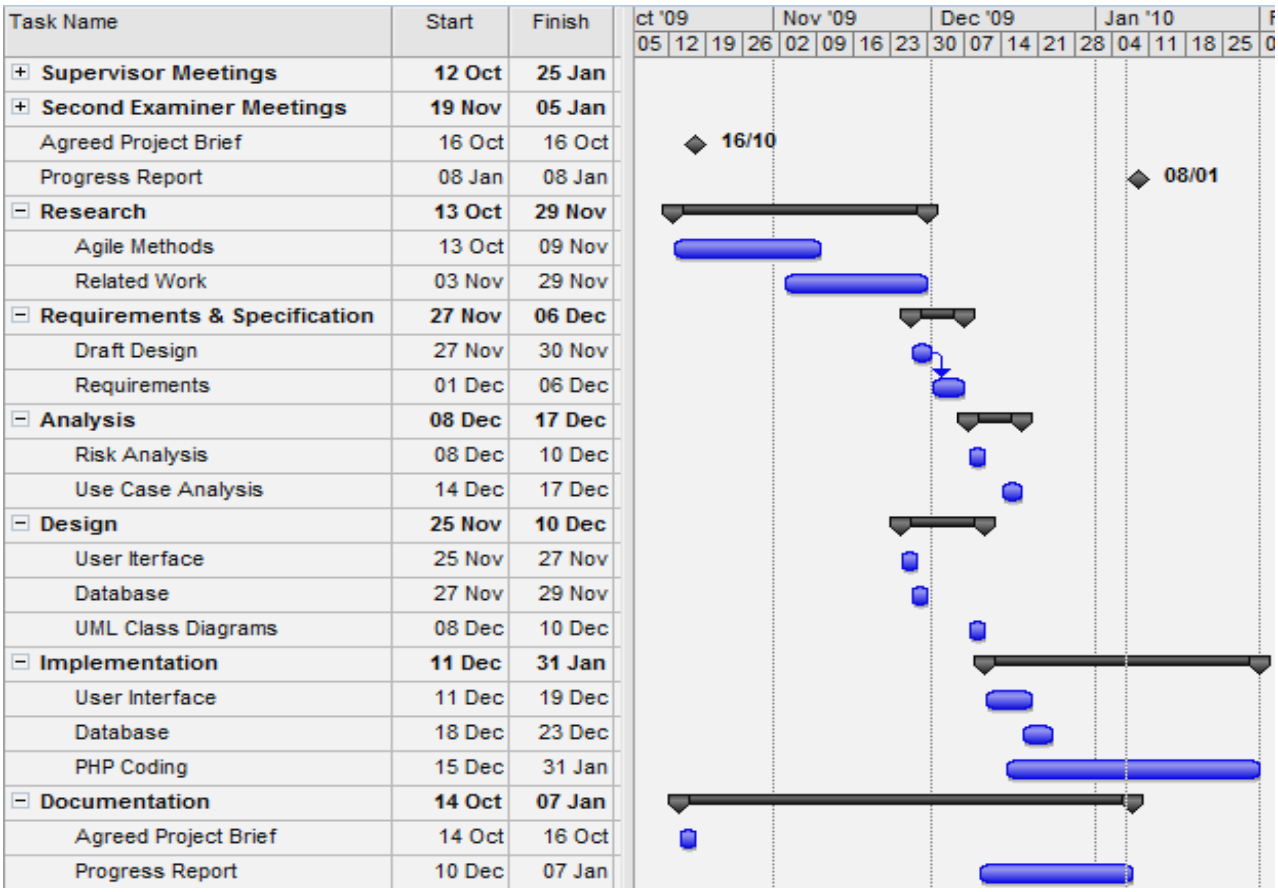

Figure B.2: Initial Semester 1 Gantt chart

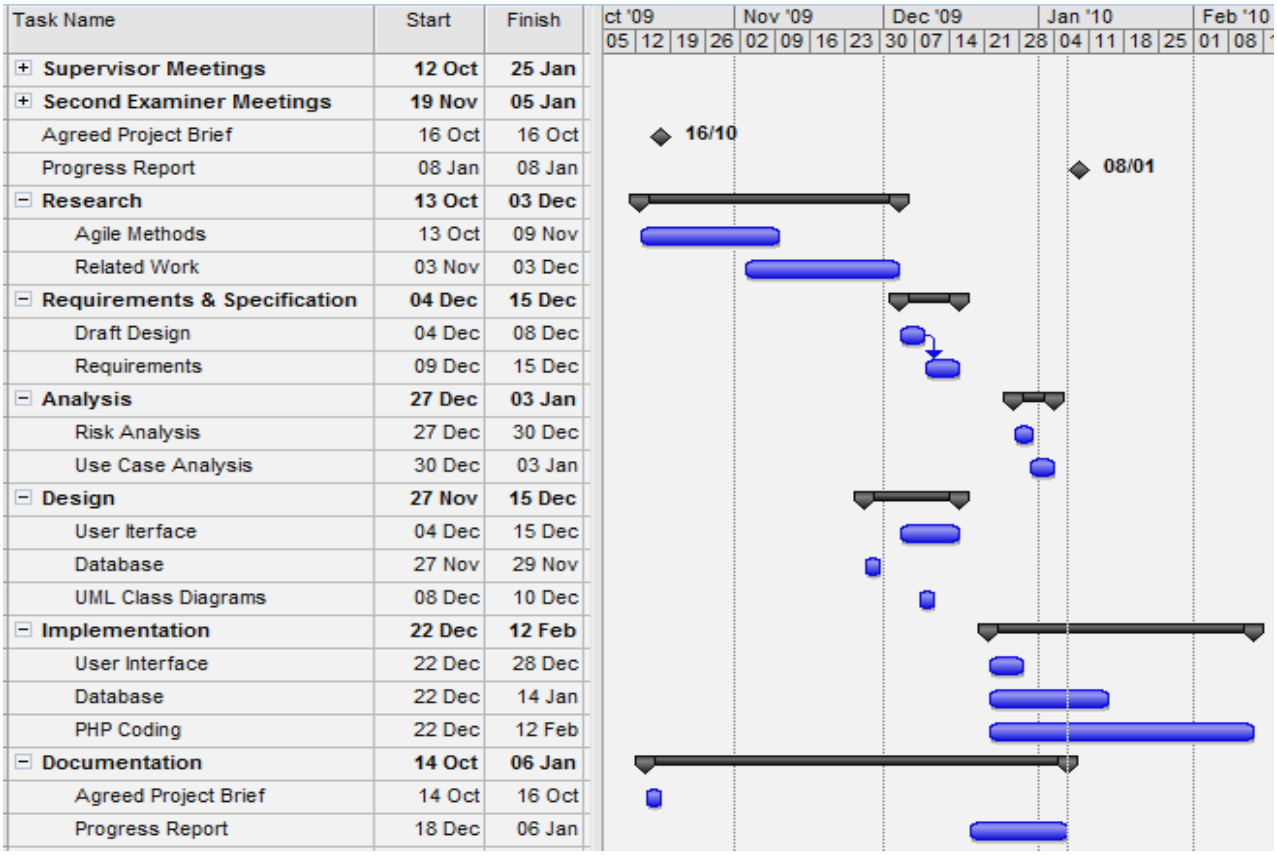

Figure B.3: Final Semester 1 Gantt chart

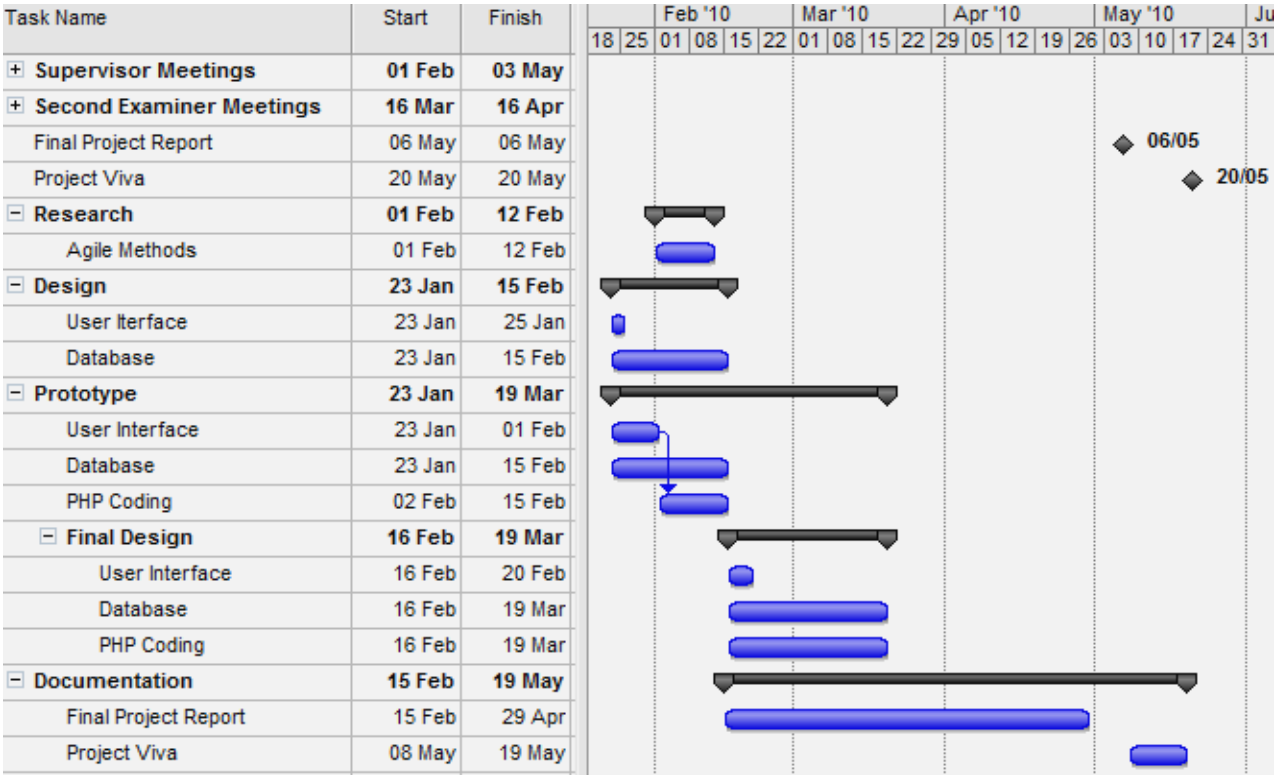

Figure B.4: Initial Semester 2 Gantt chart

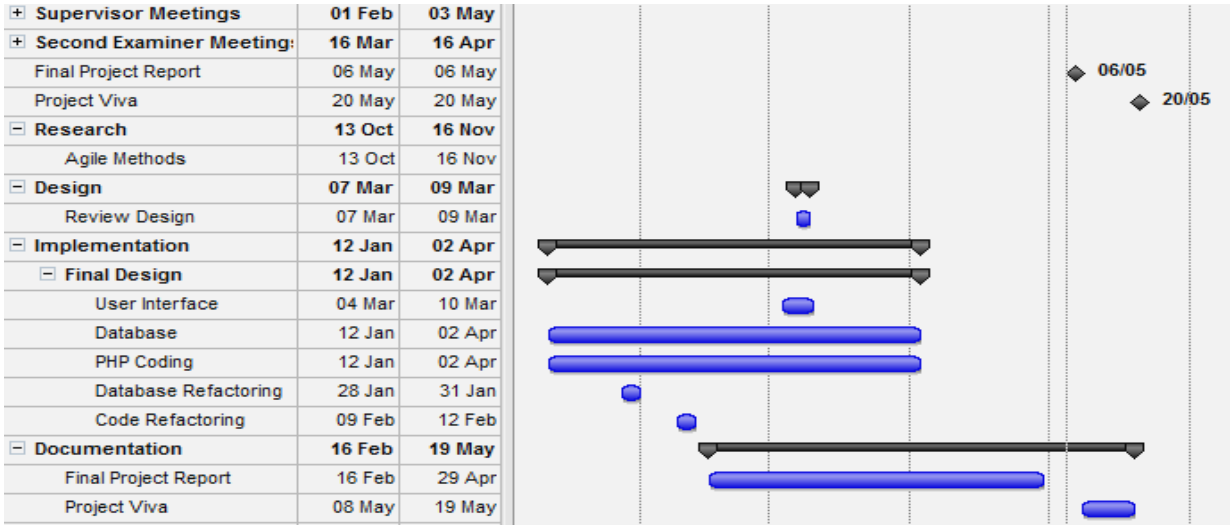

Figure B.5: Final Semester 2 Gantt chart

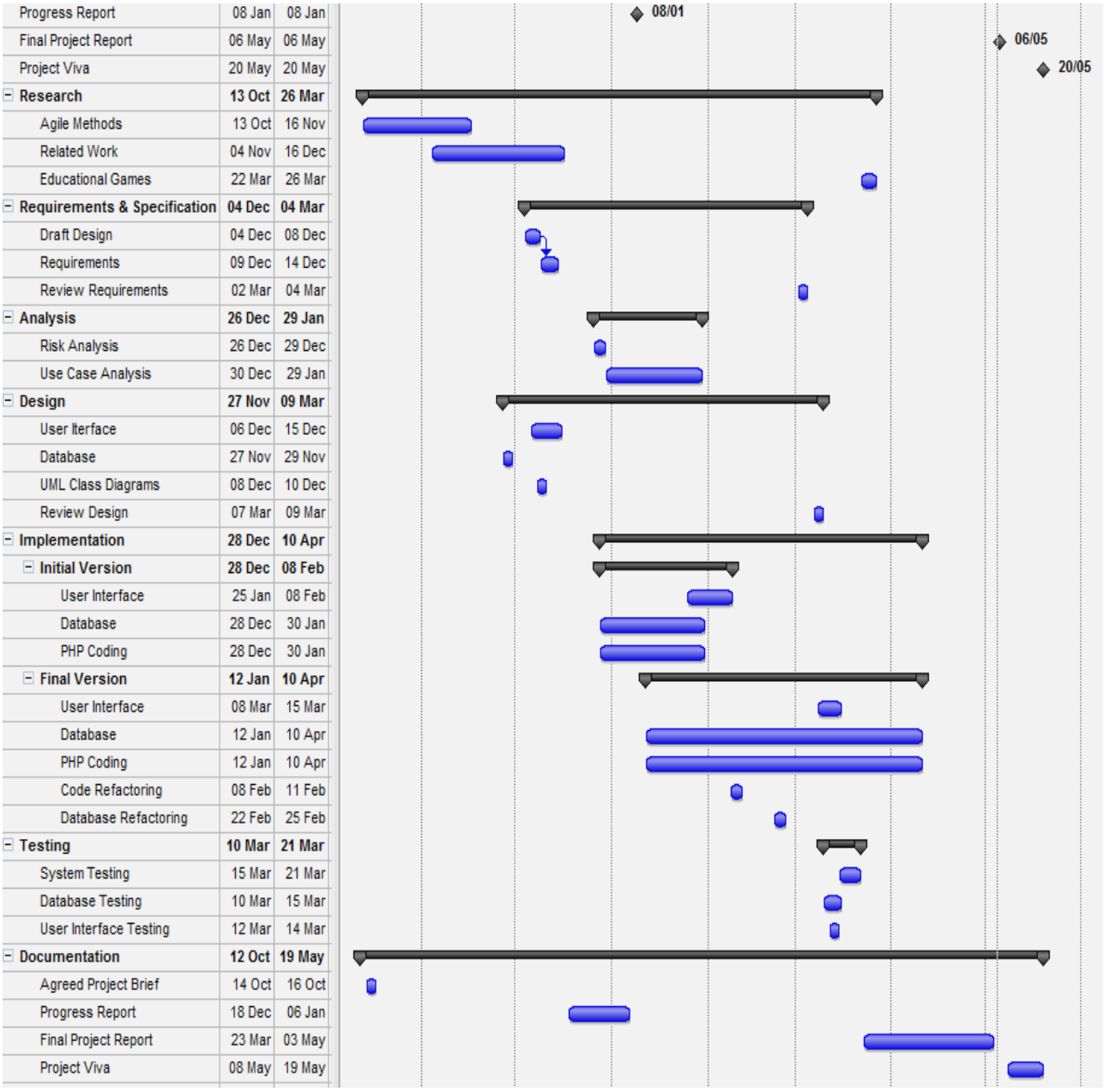

Figure B.6: Final Overall Gantt chart

From figures B.2, B.4 and B.6, it can be seen that there have been some changes in terms of the duration of tasks and the general project management. These changes occurred either because some tasks were estimated to be more difficult or easier than expected, or because the requirements have changed.

In particular, the most important change was in the duration of the actual implementation of the system, which took a week more than it was first estimated. This was because of some problems concerning information stored in the database and because the author, at the beginning of the implementation, was not as confident as at the end, so some valuable time was lost implementing features that later were considered trivial. For this reason, some of these features needed to be refactored (both in the database and the code). In addition, close to the end of the project, some of the requirements changed, so some features that had already been implemented needed to be removed or changed, and some additional functionality had to be incorporated.

The literature review took longer than expected because of the lack of availability of specific resources that could help to make the system more detailed (lack of information concerning which technique is used in which phase of the project, etc.). Also, some additional research had to be done close to the end of the project, but this did not influence the overall schedule.

During the project, there were also tasks that were completed earlier than expected such as the requirements, the design of the user interface, and the structure of the database. This allowed more time for the rest of the tasks.

In conclusion, despite the above changes to the schedule, the outcome was not influenced and the project has been delivered on time with the required quality and functionality. This is because the methodology with which the project was implemented, which allowed changes and considered them as expected during the implementation; so their impact was minor.

Appendix C – Requirements & Goals

C.1 Requirements

In this section it is possible to find the detailed requirements of the system. In particular, Table C.1 shows the functional requirements and Table C.2 the non-functional requirements. Requirements with Should priority are the requirements of the initial version and the Must-Low priority represent additional requirements for the final design.

Functional Requirements			
No	**User Story**	**Effort**	**Priority**
1	Users will be able to access the system from the Internet.	1	Must
2	Users will have to insert their username and password in order to log in to the system.	3	Must
3	New users will be required to fill in a registration form in order to add their details to the database.	3	Must
4	Passwords will be hashed to ensure security and stored in the database.	5	Must
5	Passwords cannot be viewed by other users.	5	Must
6	Usernames will be used to keep a log of the performance of each user.	5	Must
7	The username and the points that each user accumulated will be visible to all users of the system in a point-system list.	7	Must
8	New users that log in to the game for the first time, will be given a project profile with information about the project they will have to complete.	4	Must
9	For new users, a short description of what tasks they should perform is provided in order to complete the game.	4	Must
10	For returning users, the system will present the status of the project, previous deliverables and the next moves.	9	Must
11	The status of the project will contain information concerning the cost of the project up to a certain point, techniques that have been used and how effective these techniques were.	9	Must
12	The status of the project will contain information concerning the cost of the project up to a certain point, the methods and techniques that have been used and how effective these techniques were.	9	Low
13	Previous deliverables will represent the parts of the project that have been completed.	6	Must
14	Users will be able to choose which techniques they want to use for the part of the project they are in.	8	Must
15	Users will be able to choose which agile methods they want to use for the part of the project that they are in.	8	Should

Functional Requirements			
No	**User Story**	**Effort**	**Priority**
16	Users will be able to use more than one technique and agile method for a specific task	9	Should
17	Users will be able to change the method that they are using for the next deliverable.	7	Low
18	Users will be able to change the techniques that they are using for the next deliverable.	7	Should
19	Users will not be able to delete their scores.	6	Low
20	Users will not be able to delete their accounts.	5	Low
21	Scores will be a function of how appropriate a technique was for the specific phase of the project.	8	Must
22	Scores will be a function of how appropriate the combination techniques and agile methods were for the specific phase of the project.	9	Low
23	The role of the user inside the game will be the role of the manager.	-	Must
24	Users need to be able to choose one method and any of its techniques	9	Must
25	Before starting the game users will have to fill in a questionnaire concerning agile methods.	8	Must
26	After finishing the game users will have to fill in a questionnaire concerning agile methods.	8	Must

Table C.1: Functional Requirements

Non-Functional Requirements			
No	**User Story**	**Effort**	**Priority**
27	The system will be targeted at students within ECS.	5	Must
28	The system needs to function in the major browsers.	9	Low
29	The system will not require sensitive information from the user during the registration phase.	-	Must
30	The system and the database need to be secure in order to prevent attacks.	7	Should
31	The system needs to be accessible at all times.	-	Must
32	Only the administrator will be able to delete users from the database.	1	Low
33	The users of this system need to be at least second year students because some background knowledge of Software Engineering issues is assumed.	-	Must

Table C.2: Non-Functional Requirements

C.2 System Goals

1. Create a prototype of a game that simulates the use and the impact of agile methods in every phase of a project.
2. The game will be targeted at students that have already completed their first year.
3. Help students understand the phases that a project has to undergo to be delivered to the customer.
4. The delivered system must have the form of a game.
5. The game must be divided into four rounds, where each round represents a phase of the lifecycle of a project.
6. The user will have a selection of agile methods and their techniques with which they can implement their project.
7. Every choice of method and technique must be credited with a specific number of points, depending on the phase of the project.
8. The system must represent the score of the top 10 players of the game, to help competition.

C.3 Project Goals

1. Meet all the deadlines of the project.
2. Implement a project following the planned design.
3. Meet all the system goals.
4. Meet all the project goals.
5. Provide a fully functional prototype of the Agile Game.
6. Create a game that helps students understand agile methods in more depth.

C.4 Changed Requirements and Actions Taken

During the implementation the project, some of the requirements needed to change. To be more specific, the first change occurred as soon as the first version of the system was complete and the other after the completion of the evaluation questionnaire.

After the completion of the initial version of the game, the requirements and the design of the project were reviewed in order see if the outcome met the initial plan. During this process, it was found more appropriate to make clear to the users the impact that each method and its techniques have in a specific phase of the project, rather than demonstrating the way that a combination of them could affect the outcome of the project. For this reason, the initial functional requirement “Users will be able to choose a combination of agile methods to complete a task” was removed. In the final system,

users can choose only one of the agile methods and any of its techniques. This way it is easier for the user to learn the principles of each method, its different techniques and the way that these techniques can be used in every phase of the project.

Furthermore, because the system is a prototype, before the development of the final version of the game, it was considered that it would be more appropriate not to include the cost that every choice of method and technique might have on the total budget of the project. This happened because it was not possible to find specific figures on which method and which technique is more costly than others. For this reason, the initial requirement "The status of the project will contain information concerning the cost of the project until a certain point, the methods and techniques that have been used and how effective there techniques were" of the functional requirements was affected. Now, the system only contains information concerning the methods and techniques that a user has used and how effective these choices where.

For the final design, two more functional requirements (numbers 25 and 26) were added. These requirements involve the creation of a questionnaire before the start and after the completion of the game. These features were inserted in order to check the knowledge and the understanding of the user of agile methods. The first questionnaire contains general questions on agile methods. If after its completion the user gets a score lower than 50%, then they are advised to refer to the help resources provided by the system, otherwise, they can start playing the game. The second questionnaire contains slightly more difficult questions, concerning the use of the different techniques within the development process. After the completion of both questionnaires, users are able to see if their performance has improved.

Finally, the results of the evaluation questionnaire showed that users needed some more detailed feedback on why their choices were credited with the specific number of points. Also, they felt that too much prior knowledge was assumed, so for this reason more precise information was added in the help resources and in every step of the game. Another feature that was altered because of the evaluation questionnaire was the countdown clock. This feature was originally included to give the user the feeling of a real game, but some users felt that it felt more like a test, rather than a game. For this reason, the countdown clock was removed. This way, users have the opportunity to spend more time in the game, learning about agile methods while playing it. Without the clock, users will have the opportunity to refer to their notes and look for additional resources to complete the game in their own time and completing the game successfully.

Appendix D – Database Tables

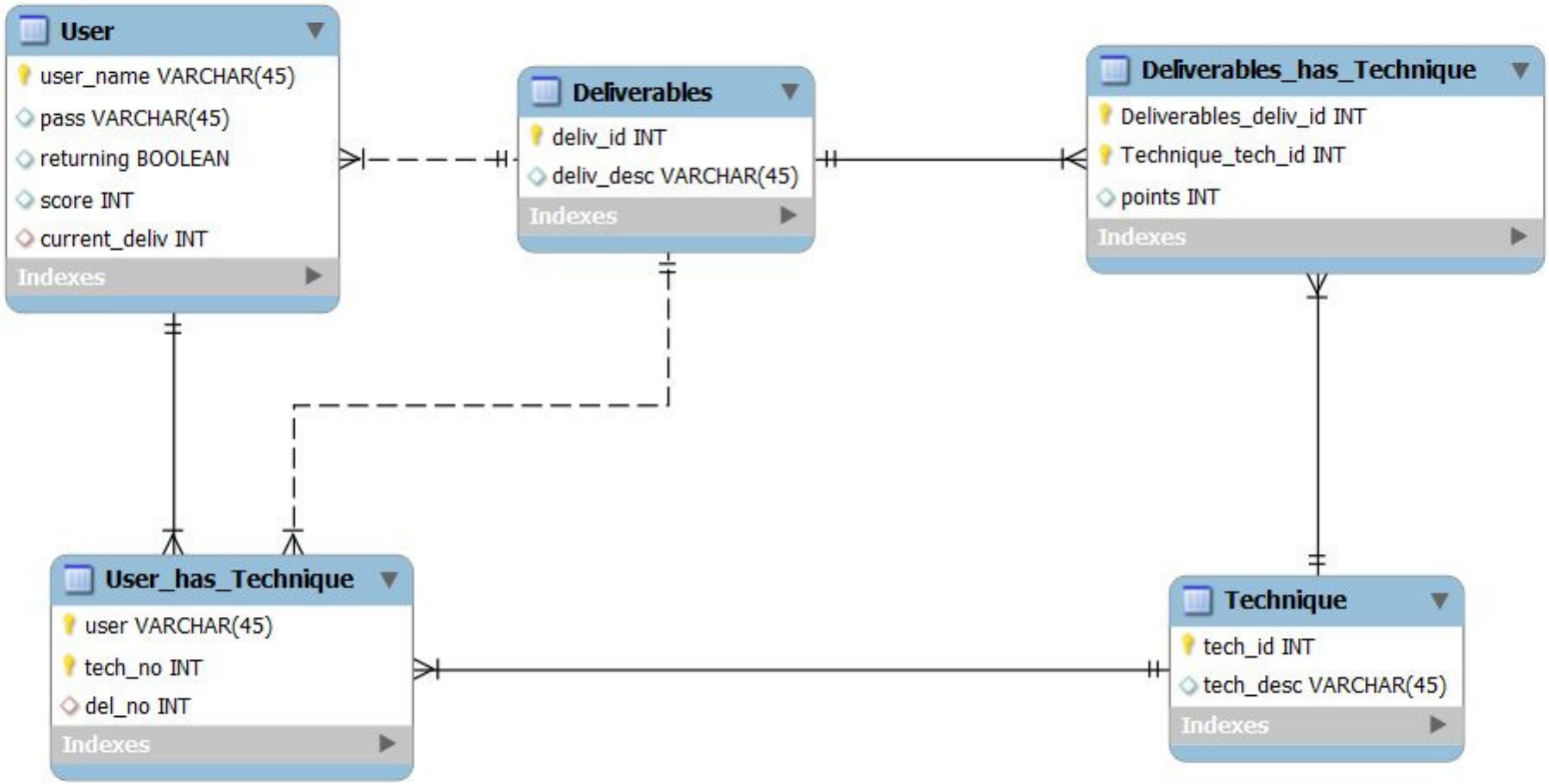

Figure D.1: Initial Database Model

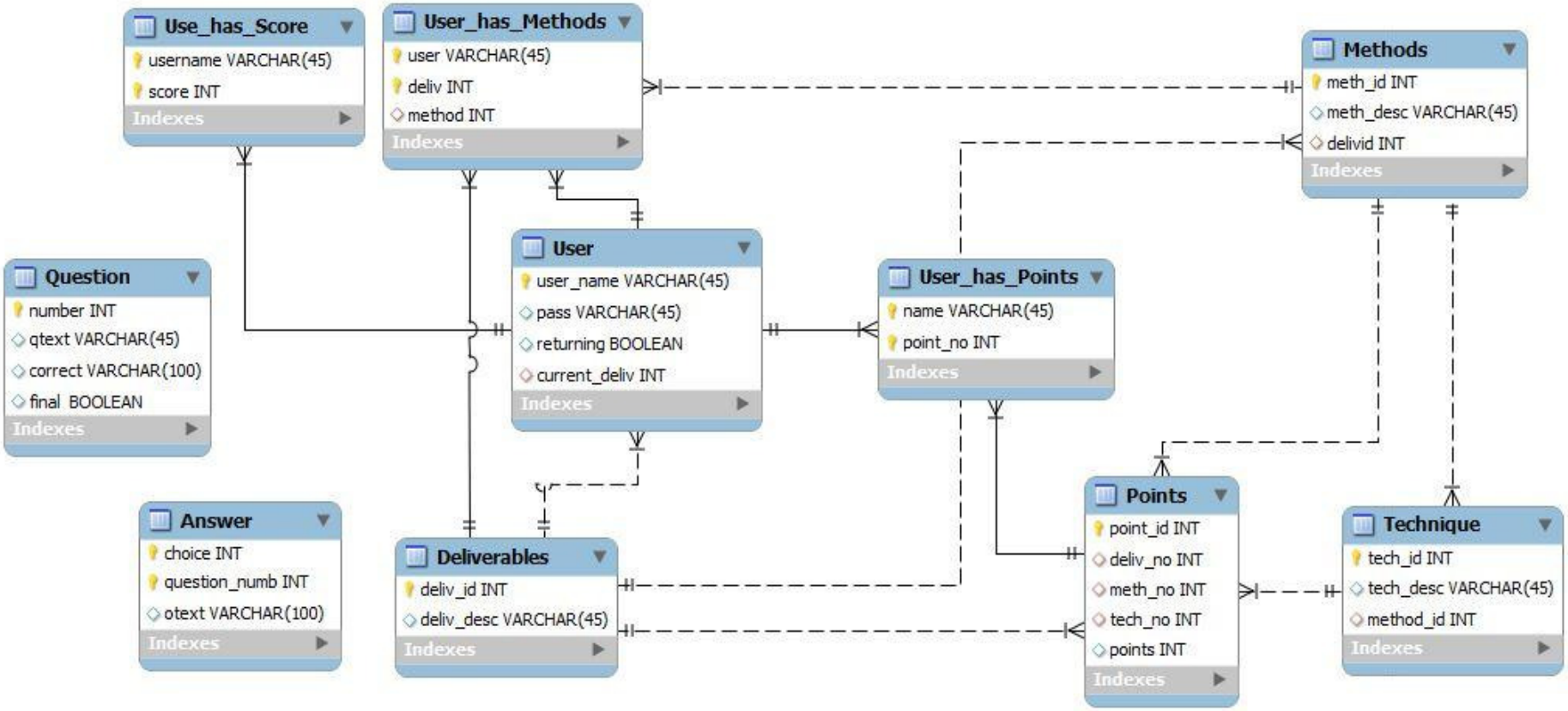

Figure D.2: Final Database Model

Appendix E – Database Schema

Below there is a detailed description about the contents of the tables in the database:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';

CREATE SCHEMA IF NOT EXISTS `db_ag2006` DEFAULT CHARACTER SET latin1
COLLATE latin1_swedish_ci ;
USE `db_ag2006`;

-- -----------------------------------------------------
-- Table `db_ag2006`.`Deliverables`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`Deliverables` ;

CREATE  TABLE IF NOT EXISTS `db_ag2006`.`Deliverables` (
  `deliv_id` INT NOT NULL ,
  `deliv_desc` VARCHAR(45) NULL ,
  PRIMARY KEY (`deliv_id`) )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`User`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`User` ;

CREATE  TABLE IF NOT EXISTS `db_ag2006`.`User` (
  `user_name` VARCHAR(45) NOT NULL ,
  `pass` VARCHAR(45) NULL ,
  `returning` BOOLEAN NULL DEFAULT 0 ,
  `current_deliv` INT NULL ,
  PRIMARY KEY (`user_name`) ,
  INDEX `current_deliv` (`current_deliv` ASC) ,
  CONSTRAINT `current_deliv`
    FOREIGN KEY (`current_deliv` )
    REFERENCES `db_ag2006`.`Deliverables` (`deliv_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`Methods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`Methods` ;
```

```
CREATE  TABLE IF NOT EXISTS `db_ag2006`.`Methods` (
  `meth_id` INT NOT NULL ,
  `meth_desc` VARCHAR(45) NULL ,
  `delivid` INT NULL ,
  PRIMARY KEY (`meth_id`) ,
  INDEX `delivid` (`delivid` ASC) ,
  CONSTRAINT `delivid`
    FOREIGN KEY (`delivid` )
    REFERENCES `db_ag2006`.`Deliverables` (`deliv_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`Technique`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`Technique` ;

CREATE  TABLE IF NOT EXISTS `db_ag2006`.`Technique` (
  `tech_id` INT NOT NULL ,
  `tech_desc` VARCHAR(45) NULL ,
  `method_id` INT NULL ,
  PRIMARY KEY (`tech_id`) ,
  INDEX `method_id` (`method_id` ASC) ,
  CONSTRAINT `method_id`
    FOREIGN KEY (`method_id` )
    REFERENCES `db_ag2006`.`Methods` (`meth_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`User_has_Methods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`User_has_Methods` ;

CREATE  TABLE IF NOT EXISTS `db_ag2006`.`User_has_Methods` (
  `user` VARCHAR(45) NOT NULL ,
  `deliv` INT NOT NULL ,
  `method` INT NULL ,
  PRIMARY KEY (`user`, `deliv`) ,
  INDEX `user` (`user` ASC) ,
  INDEX `deliv` (`deliv` ASC) ,
  INDEX `method` (`method` ASC) ,
  CONSTRAINT `user`
    FOREIGN KEY (`user` )
    REFERENCES `db_ag2006`.`User` (`user_name` )
    ON DELETE NO ACTION
```

```
    ON UPDATE NO ACTION,
  CONSTRAINT `deliv`
    FOREIGN KEY (`deliv` )
    REFERENCES `db_ag2006`.`Deliverables` (`deliv_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `method`
    FOREIGN KEY (`method` )
    REFERENCES `db_ag2006`.`Methods` (`meth_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`Points`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`Points` ;

CREATE  TABLE IF NOT EXISTS `db_ag2006`.`Points` (
  `point_id` INT NOT NULL ,
  `deliv_no` INT NOT NULL ,
  `meth_no` INT NOT NULL ,
  `tech_no` INT NOT NULL ,
  `points` INT NULL ,
  INDEX `meth_no` (`meth_no` ASC) ,
  INDEX `tech_no` (`tech_no` ASC) ,
  INDEX `deliv_no` (`deliv_no` ASC) ,
  PRIMARY KEY (`point_id`) ,
  CONSTRAINT `meth_no`
    FOREIGN KEY (`meth_no` )
    REFERENCES `db_ag2006`.`Methods` (`meth_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `tech_no`
    FOREIGN KEY (`tech_no` )
    REFERENCES `db_ag2006`.`Technique` (`tech_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `deliv_no`
    FOREIGN KEY (`deliv_no` )
    REFERENCES `db_ag2006`.`Deliverables` (`deliv_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`User_has_Points`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`User_has_Points` ;
```

```
CREATE  TABLE IF NOT EXISTS `db_ag2006`.`User_has_Points` (
  `name` VARCHAR(45) NOT NULL ,
  `point_no` INT NOT NULL ,
  PRIMARY KEY (`name`, `point_no`) ,
  INDEX `user` (`name` ASC) ,
  INDEX `point_no` (`point_no` ASC) ,
  CONSTRAINT `user`
    FOREIGN KEY (`name` )
    REFERENCES `db_ag2006`.`User` (`user_name` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `point_no`
    FOREIGN KEY (`point_no` )
    REFERENCES `db_ag2006`.`Points` (`point_id` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `db_ag2006`.`Question`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `db_ag2006`.`Question` ;

CREATE TABLE IF NOT EXISTS `db_ag2006`.`Question` (
  `number` INT NOT NULL ,
  `qtext` VARCHAR(100) NULL,
  `correct` VARCHAR(100) NULL,
  `final` INT NULL(11) ,
  PRIMARY KEY (`number`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `db_ag2006`.`Answer`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `db_ag2006`.`Answer` ;


CREATE TABLE IF NOT EXISTS `db_ag2006`.`Answer` (
  `choice` INT(11) NOT NULL ,
  `question_numb` INT NOT NULL,
  `otext` VARCHAR(100) NULL,
  `stage` INT(11) NULL
  PRIMARY KEY (`choice`,`question_numb`),
  INDEX `fk_Answer_Question` ( `question_numb` ASC),
  CONSTRAINT `fk_Answer_Question`
    FOREIGN KEY (`question_numb`)
    REFERENCES `db_ag2006`.`Question`(`number`)
    ON DELETE CASCADE
```

```
   ON UPDATE CASCADE )
ENGINE = InnoDB;

-- -----------------------------------------------------
-- Table `db_ag2006`.`User_has_Score`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `db_ag2006`.`User_has_Score` ;

CREATE TABLE IF NOT EXISTS `db_ag2006`.`User_has_Score` (
  `username` VARCHAR(45) NOT NULL,
  `init_score` INT NOT NULL,
  `final_score` INT NOT NULL,
  PRIMARY KEY (`username`,`init_score`,`final_score`),
  INDEX `username`(`username` ASC),
  INDEX `init_score` (`init_score` ASC),
  INDEX `final_score` (`final_score` ASC),
  CONSTRAINT `username`
    FOREIGN KEY (`username`)
    REFERENCES `db_ag2006`.`User` (`user_name`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Appendix F – Test Cases

Below there are the tests that were conducted in order to check the functionality of the system:

F.1 System Testing - Black Box Testing

No	Test Case	Expected Outcome	Result
1	Users access the homepage of the system.	Homepage is loaded.	Pass
2	Users click “Log In” button and log in to the system without being registered.	Notification that they are not logged in and redirects them to the homepage.	Pass
3	Users log in without entering a username.	Pop-up message that user id is blank appears.	Pass
4	Users log in without entering a password.	Pop-up message that password is blank appears.	Pass
5	Users click on “Register” link.	Page register.php is loaded.	Pass
6	Users insert valid username and password to register.	User details inserted in the database.	Pass
7	Users insert only their username to register.	Pop-up message that password is blank appears.	Pass
8	Users insert only their username to register.	Details have not been recorded by the system.	Pass
9	Users insert only their password to register.	Pop-up message that username is blank appears.	Pass
10	Users insert only their password to register.	Details have not been recorded by the system.	Pass
11	Users leave all fields of register form blank.	Pop-up message that details are missing appears.	Pass
12	Users leave all fields of register form blank.	Details have not been recorded by the system.	Pass
13	While users type a username, the system checks username availability.	Message about availability appears while users type next to username textbox.	Pass
14	Users insert usernames fewer than 3 characters long to register.	Pop-up message that username should be at least 3 characters long appears.	Pass
15	Username contains characters different from [^A-Za-z0-9_-] to register.	Pop-up message that only letters, numbers, _, - and ^ characters are allowed appears.	Pass
16	Users insert valid username and password to register.	Users registered and notification that they can now log in.	Pass
17	Users click on “Log In” link.	Users are redirected in the homepage.	Pass
18	Users click on “Log In” button leaving the login form blank.	Pop-up message that details are missing appears.	Pass

No	Test Case	Expected Outcome	Result
19	Users insert only their username to log in.	Pop-up message that password is blank appears.	Pass
20	Users insert only their password to log in.	Pop-up message that username is blank appears.	Pass
21	Users insert usernames fewer than 3 characters long to register.	Pop-up message that username should be at least 3 characters long appears.	Pass
22	Username contains characters different from [^A-Za-z0-9_-] to register.	Pop-up message that only letters, numbers, _, - and ^ characters are allowed appears.	Pass
23	Users insert valid username and password to log in.	Users are redirected to introduction.php if first time users.	Pass
24	Display the username of player in introduction.php.	Display “Hello <username>”.	Pass
25	Users click on “Log Out” button to exit the game.	Users exit the game and are redirected in the home page.	Pass
26	Users log in again to the system.	Users are redirected in main.php.	Pass
27	Display the name of the player in main.php.	Display “Welcome back <username>.	Pass
28	Display the username of the player in main.php.	Display “You are currently on deliverable <deliverable_number>.	Pass
29	Users click on “Log Out” while in introduction.php and log back in again.	Users are redirected in questionnaire.php.	Fail
30	In questionnaire.php CSS is loaded.	Background picture appears.	Pass
31	Questions and answers appear in questionnaire.php.	All questions and answers appear.	Pass
32	One radio button of each question is pre-checked.	Bottom radio button of each question is pre-checked.	Pass
33	Users click on “Clear” button in questionnaire.php.	All radio buttons are reset to the default position.	Pass
34	Users click on “Check Results” button in questionnaire.php.	Pop-up message to proceed appears.	Pass
35	Pop-up message gives users the option to either proceed or cancel.	Pop-up message provided “OK” and “Cancel” options.	Pass
36	Users select “OK” option in pop-up message.	Users are redirected to results.php to view their score.	Pass
37	Users select “Cancel” option in pop-up message.	Users return to questionnaire.php to alter their options.	Pass
38	Page results.php provides feedback on players’ answers.	Information on which questions were wrong (if any) and why and the overall score.	Pass
39	Users score less that 50% on the questionnaire.	“Help” button appears.	Pass
40	Users click on “Proceed” button on results.php.	Users are redirected to round.php.	Pass

No	Test Case	Expected Outcome	Result
41	Users click “Submit” button without making any selections.	Error message appears “You haven’t selected a method or techniques” and “Proceed” button takes users to round.php.	Pass
42	Users select one method and then click on “Submit” button.	Error message appears “You haven’t selected a method or techniques” and “Proceed” button takes users to round.php.	Pass
43	Users select one method and some of its techniques and then check another method and some of its techniques without un-checking the techniques of the first method.	The score of the player at this stage is the addition of the points of the techniques of the currently selected method (ignoring the techniques of the first selected method).	Pass
44	Users make their selections correctly and click on the “Submit” button.	Users are redirected in report.php to view detailed feedback.	Pass
45	Users click on “Proceed” button while in report.php.	Users are redirected to round.php for the next deliverable.	Pass
46	Users reach finish deliverable 4, are in report.php and click “Proceed” button.	Users are redirected to the second questionnaire.	Pass
47	Users have completed the second questionnaire and are in results.php and click “Proceed” button.	Users are redirected to summary.php where they can view of their progress and the score of the top 10 players.	Pass

Table F.1: System Testing – Black Box Testing

F.2 Database Testing

No	Test Case	Expected Outcome	Result
48	Users insert a username that exceeds 45 characters.	The database stores only the first 45 characters of the username.	Pass
49	Users insert a password that exceeds 45 characters.	The database stores only the first 45 characters of the password.	Pass
50	Users leave the username field empty.	JavaScript pops-up a message notifying that username is blank.	Pass
51	Users leave the password field empty.	JavaScript pops-up a message notifying that password is blank.	Pass
52	Users leave username or password fields empty.	If JavaScript is disabled, blank fields are not inserted in the database.	Pass

Table F.2: Database Testing

F.3 Functional Requirements Testing

No	Test Case	Expected Outcome	Result
53	Users will be able to access the system from the Internet.	Homepage is loaded.	Pass
54	Users will have to insert their username and password to log in to the system.	If correct username and password, users log in the system.	Pass
55	New users will be required to fill in a registration form to add their details in the database.	The registration form is loaded, users insert their correct details and the information is stored in the database.	Pass
56	Passwords will be hashed to ensure security and stored in the database.	Passwords are stored hashed in the database using `sha()` function.	Pass
57	Passwords cannot be viewed by other users.	Passwords are represented as dots while users type their passwords	Pass
58	Usernames will be used to keep a log on the performance of each user.	Usernames will be used to represent the overall score of the user in the game.	Pass
59	The username and the points that each user accomplished will be visible to all users of the system in a point-system list.	The username and the score of the top 10 players are represented as a high score board.	Pass
60	New users that log in to the game for the first time, will be given a project profile with information about the project they will have to complete.	First time users, after registering and logging in are redirected in introduction.php which contains the project profile.	Pass
61	For new users, a short description of what tasks they should perform is provided in order to complete the game.	First time users, after registering and logging in are redirected in introduction.php which contains the project profile.	Pass
62	For returning users, the system will present the status of the project, previous deliverables and the next moves.	Returning users, after logging in to the system, are redirected to main.php which contains a summary of the users' progress until this point.	Pass
63	The status of the project will contain information concerning the techniques that have been used and how effective these techniques were.	The main.php contains information about the methods and techniques that players used and their overall score.	Pass
64	The status of the project will contain information concerning the methods and techniques that have been used and how effective these techniques were.	The main.php contains information about the methods and techniques that players used and their overall score.	Pass
65	Previous deliverables will	Every deliverable corresponds to a	Pass

No	Test Case	Expected Outcome	Result
	represent the parts of the project that have been completed.	project lifecycle.	
66	Users will be able to choose which techniques they want to use for the part of the project they are in.	Users are able to choose a number of techniques.	Pass
67	Users will be able to choose which agile methods they want to use for the part of the project that they are in.	The system offers a number of agile methods and users can pick one them.	Pass
68	Users will be able to use more than one technique and agile method for a specific task.	Users are able to choose a number of techniques.	Pass
69	Users will be able to change the method that they are using for the next deliverable.	Proceeding to the next deliverable users can choose one of the methods independently of their previous choices.	Pass
70	Users will be able to change the techniques that they are using for the next deliverable.	Proceeding to the next deliverable, users can choose one of the methods and any of the techniques independently of their previous choices.	Pass
71	Users will not be able to delete their scores.	Only the administrator has access the database.	Pass
72	Users will not be able to delete their accounts.	Only the administrator has access the database.	Pass
73	Scores will be a function of how appropriate a technique was for the specific phase of the project.	Score is dependent on how appropriate the chosen techniques are for the specific phase of the project.	Pass
74	Scores will be a function of how appropriate the combination techniques and agile methods were for the specific phase of the project.	Score is dependent on how appropriate a chosen method and its techniques are for the specific phase of the project.	Pass
75	The role of the user inside the game will be the role of the manager.	Users hold the role of the Project Manager of the game.	Pass
76	Users need to choose one method and any of its techniques.	Users choose one method and any of its techniques.	Pass
77	Before starting the game users need to fill in a questionnaire concerning agile methods.	A questionnaire for agile methods was created.	Pass
78	After finishing the game users need to fill in a questionnaire concerning agile methods.	A questionnaire for agile methods was created.	Pass

Table F.3: Functional Requirements Testing

F.4 Non-Functional Requirements Testing

No	Test Case	Expected Outcome	Result
79	The system will be targeted at students within ECS.	The system is targeted at ECS students.	Pass
80	The system needs to function in the majority of browsers.	The system functions on Mozilla Firefox and Internet Explorer.	Fail
81	The system will not require sensitive information from the user during the registration phase.	The system requires only a username and a password.	Pass
82	The system and the database of the system need to be secure in order to prevent attacks.	Sessions and prevention against SQL injection was used.	Pass
83	The system needs to be accessible at all times.	The system is stored on the ECS server.	Pass
84	Only the administrator will be able to delete users from the database.	Only the administrator of the system has access to the database.	Pass
85	The users of this system need to be at least second year students because some background knowledge on Software Engineering issues is assumed.	The users of the system were at least second year students on Computer Science.	Pass

Table F.4: Non-Functional Requirements Testing

As mentioned in Section 6.2 only 2.3% of the tests failed. In particular, only 2 out of 85 different test cases failed. The first test was test case 29 in the system testing section which was "Users click on "Log Out" while in introduction.php and log in back again." This bug could have been fixed, but due to time constraints the author was unable to fix it. The other test that failed was test case 80 in the Non-Functional requirements testing section which was "The system needs to function in the majority of browsers". The specific requirement was a low priority requirement (see Table C.2) and because its effort rate was very high, it was considered right to focus only on two of the available browsers (Mozilla Firefox and Internet Explorer) rather than trying to comply the interface with the requirements of every browser.

The two test cases that failed, do not affect the overall functionality and the final aim of the project, so for this reason the project can be considered as successful in terms of the test cases.

Appendix G – Evaluation Questionnaire

Below there is the evaluation questionnaire that users completed to evaluate the Agile Game. The analytical results can be found on Appendix H.

Background

1. What is your year of study?
    a. 1st year
    b. 2nd year
    c. 3rd year
    d. 4th year
    e. MSc

2. Are you aware of any of the following agile methods?
    a. XP Programming ☐
    b. Scrum ☐
    c. Crystal ☐
    d. Feature Driven Development (FDD) ☐
    e. Dynamic Systems Development Method (DSDM) ☐

3. Are you familiar with any of the following agile techniques?
    a. Pair programming ☐
    b. Code refactoring ☐
    c. Continuous integration ☐
    d. Test Driven Design (TDD) ☐
    e. Stakeholder Participation ☐
    f. Code regression testing ☐
    g. Daily scrum meeting ☐
    h. Sprint review meeting ☐
    i. Sprint planning meeting ☐
    j. Product backlog ☐
    k. Sprint backlog ☐
    l. Burndown chart ☐

4. Do you believe that you are more familiar with traditional project management methods (e.g. Waterfall model, Spiral model etc) rather than agile methods?
    a. Yes
    b. No
    c. I am familiar with both

5. Have you been taught about agile methods whilst being at University?
    a. Yes
    b. No
    c. No, by work experience

(Skip question 6 if you choose option c.)

6. What resources did you use to learn about agile methods during your course?
    a. Text books
    b. Lecture notes
    c. Online resources

Game Usability

7. Too much prior knowledge was assumed:
    I strongly disagree ○ ○ ○ ○ ○ I strongly agree

8. The user interaction with the system was smooth:
    I strongly disagree ○ ○ ○ ○ ○ I strongly agree

9. Was the interface of the game pleasant?
    I strongly disagree ○ ○ ○ ○ ○ I strongly agree

10. Would you like to add any comments about the user interface of the game?

Learning

11. The game helps the user understand the use of agile methods and fulfils its educational aim:
    I strongly disagree ○ ○ ○ ○ ○ I strongly agree

12. Did the system provide you with a satisfying amount of feedback in every step of the game?
    I strongly disagree ○ ○ ○ ○ ○ I strongly agree

13. In what ways could the system be improved to excel the users' understanding on agile methods?

14. Did you understand the different techniques of every method in more depth through the Agile Game?
    a. Yes
    b. No
    c. It did not become very clear

15. Were the help resources useful and informative?

I strongly disagree ○ ○ ○ ○ ○ I strongly agree

16. Which method did you understand in more depth after playing the Agile Game?
    a. XP Programming ☐
    b. Scrum ☐
    c. Crystal ☐
    d. Feature Driven Development (FDD) ☐
    e. Dynamic Systems Development Method (DSDM) ☐

17. Out of the following techniques with which one do you feel more confident after playing the game?
    a. Pair programming ☐
    b. Code refactoring ☐
    c. Continuous integration ☐
    d. Test Driven Design (TDD) ☐
    e. Stakeholder Participation ☐
    f. Code regression testing ☐
    g. Daily scrum meeting ☐
    h. Sprint review meeting ☐
    i. Sprint planning meeting ☐
    j. Product backlog ☐
    k. Sprint backlog ☐
    l. Burndown chart ☐

18. Was it clear in which phase of the project lifecycle is more appropriate to use each method and technique?
    a. Yes

b. No

19. If no, why?

System

20. How would you rate the system overall?

Very poor ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Excellent

21. Would you like to add any more comments about the overall system?

Appendix H – Evaluation Questionnaire & Interview Results

Below there are analytic graphs demonstrating the results of the evaluation questionnaire:

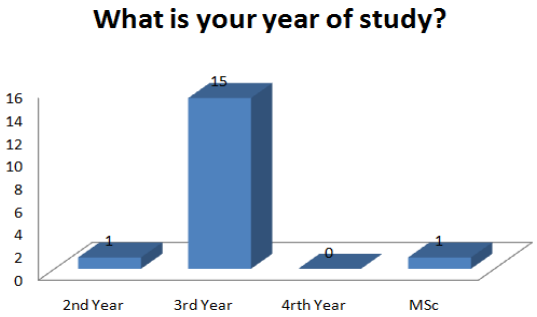

Figure H.1: Question 1

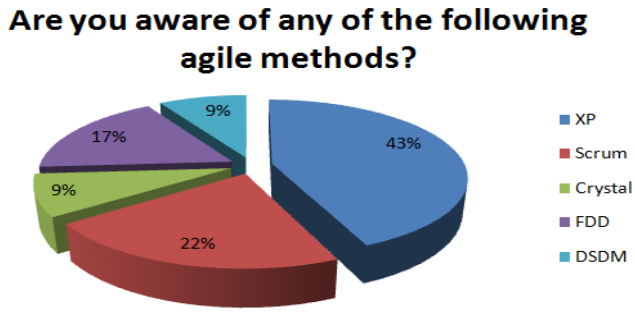

Figure H.2: Question 2

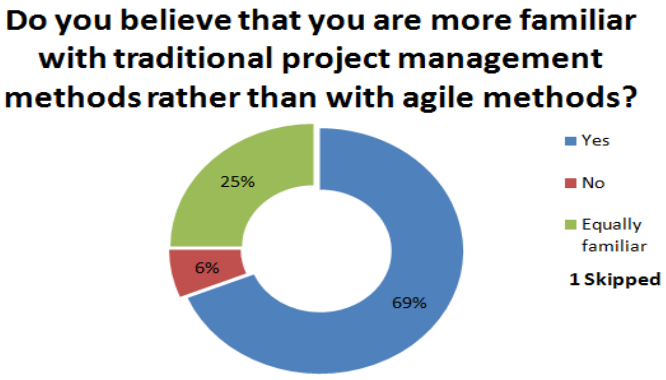

Figure H.3: Question 3

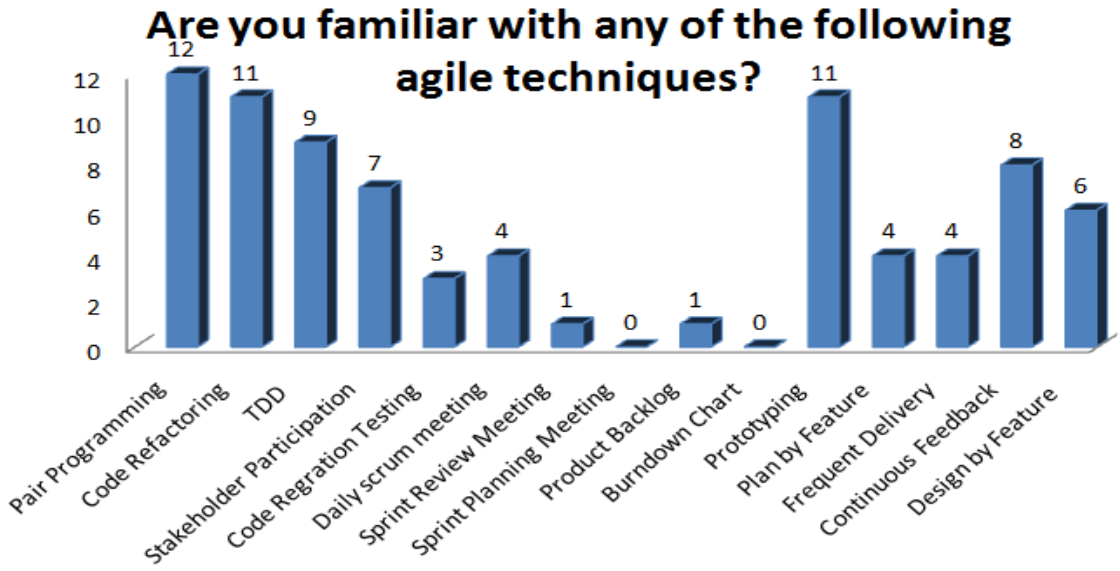

Figure H.4: Question 4

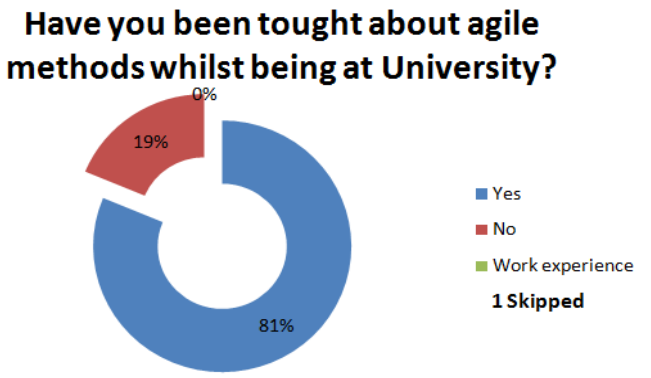

Figure H.5: Question 5

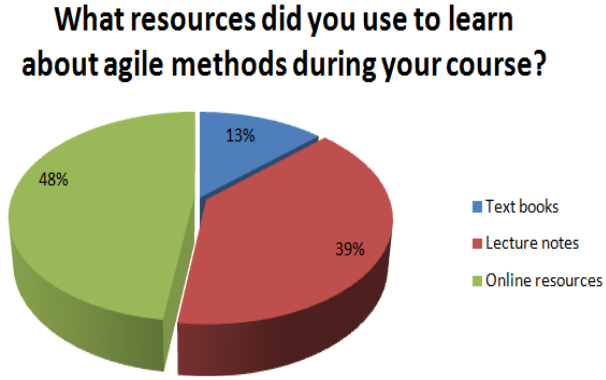

Figure H.6: Question 6

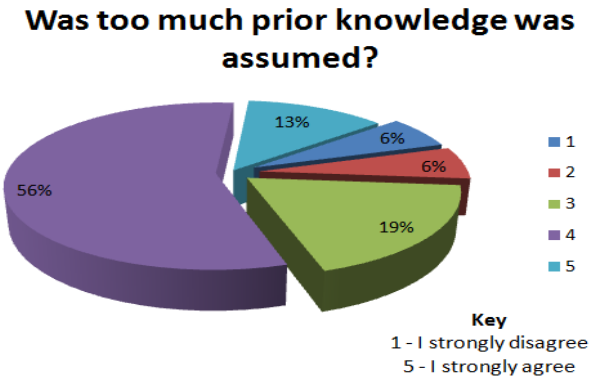

Figure H.7: Question 7

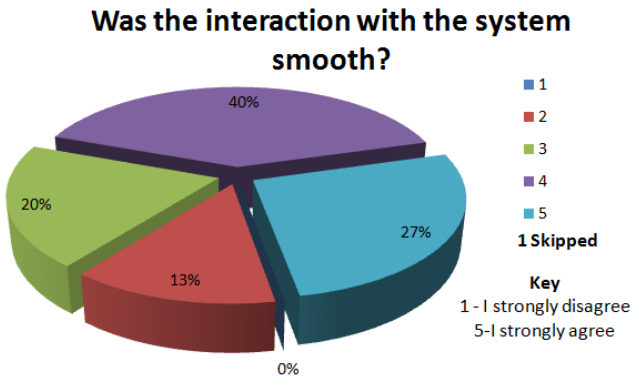

Figure H.8: Question 8

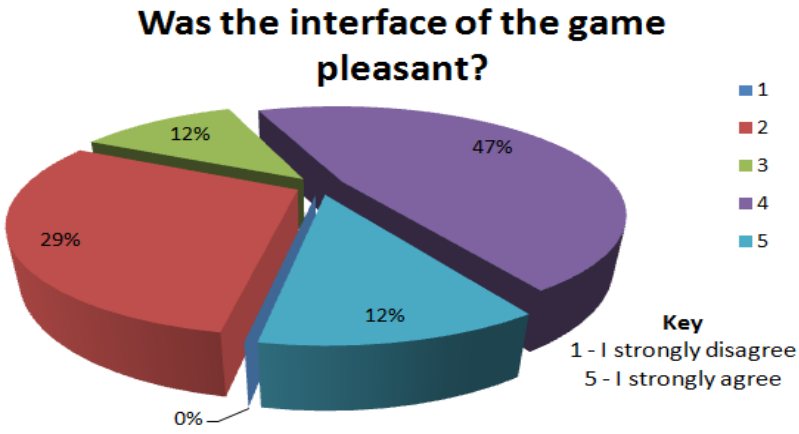

Figure H.9: Question 9

Would you like to add any comments about the user interface of the game?

Use nicer background picture. :)
The background (the one that looks like a notepad) was a bit too narrow for my screen resolution (1366x768), so some text was sticking out.
The fonts were huge!
The interface of the page with the notebook background appears misplaced. On another page upon selecting the method i get the message "You have not selected a method or techniques" I can not move any further.
User interface is clear and easy to use.
Very simple
When choosing options during the game, could make the other sub-menus disappear, making it easier to find the options you want.
i would say it was very messy!
The idea was nice, the design could have been a bit better

Figure H.10: Question 10

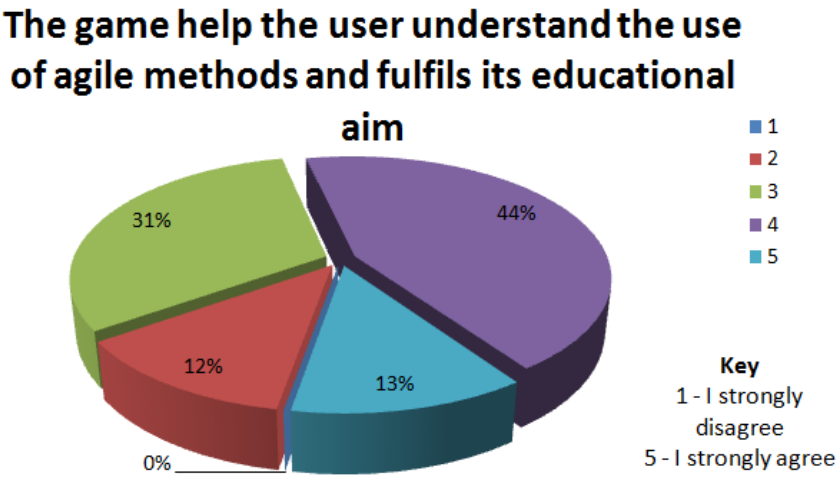

Figure H.11: Question 11

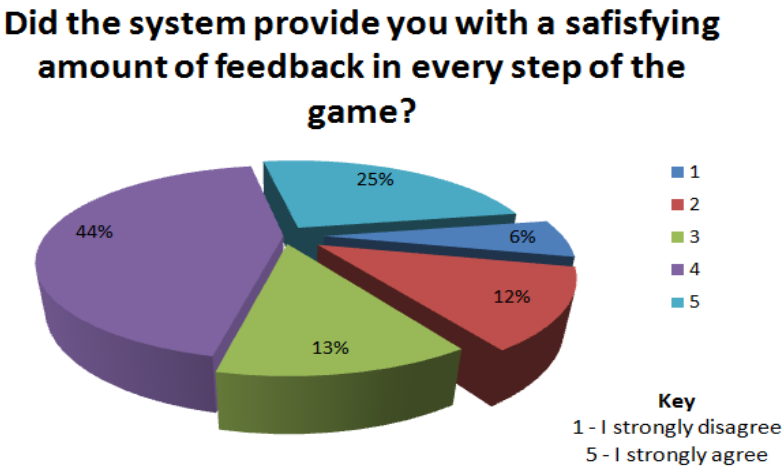

Figure H.12: Question 12

13. In what ways could the system be improved to excel the users' understanding on agile methods?

May be be more interactive. Rather than just answering question, you can play the game where you run developers team. A goal would be to run the team as effectively as possible...

Give a reason for the mistakes made and a short explanation.

Provide more descriptions and feedback during the questions rather than just after they are completed. There should be some more descriptions of the situations / principles as well as a more detailed description of what was right / wrong with the answers.

The game is basically a test, that gives feedback when a user finishes it. If someone doesn't have any knowledge of some techniques, it is difficult to select the correct answers, and therefore is a frustrating experience. I would add some teaching material in the beginning, for those who doesn't know anything.

Short textual descriptions of the techniques or methods would help in learning during gameplay.

It has already a good help section. I think that this seciton could be organized in a better way.

A little more detailed or explanatory feedback may be more helpful during the game. Perhaps include what the correct answers might have been.

The system could give out marks on each correct answer so the student will have the filling of a competition that would make him or her do well.

More precise feedback

some information at the beginning about what exactly are agile methods and maybe an explanation of what each approach aims for.

More meaningful feedback

Figure H.13: Question 13

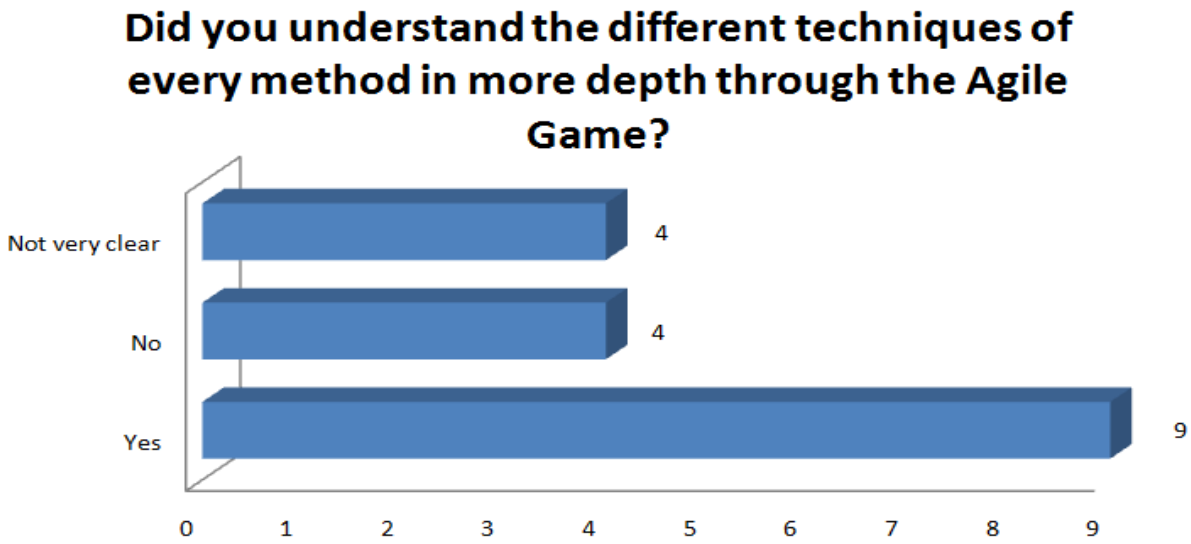

Figure H.14: Question 14

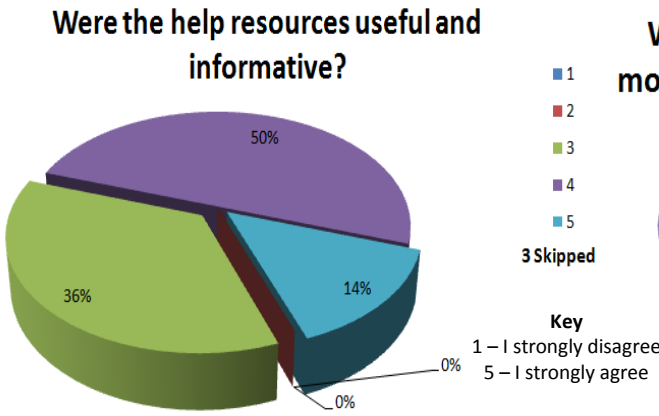

Figure H.15: Question 15

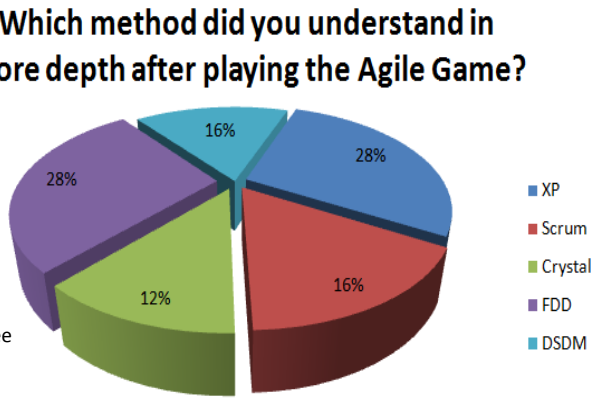

Figure H.16: Question 16

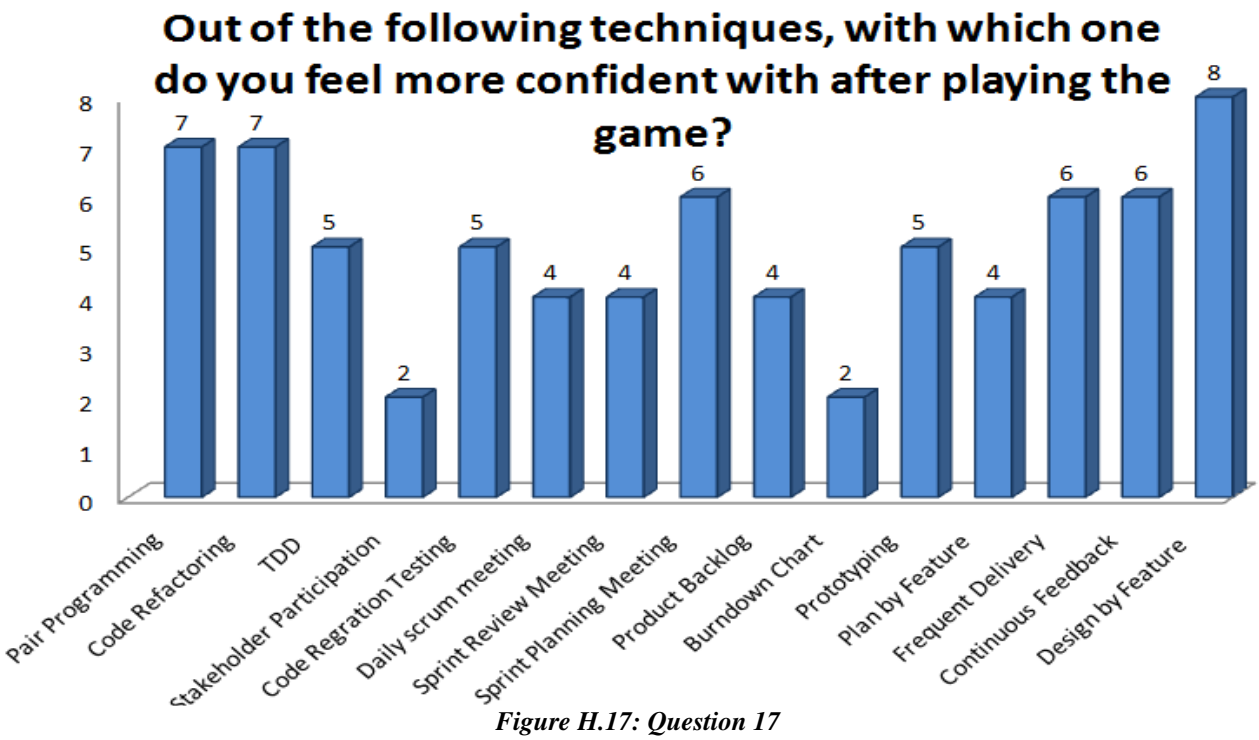

Figure H.17: Question 17

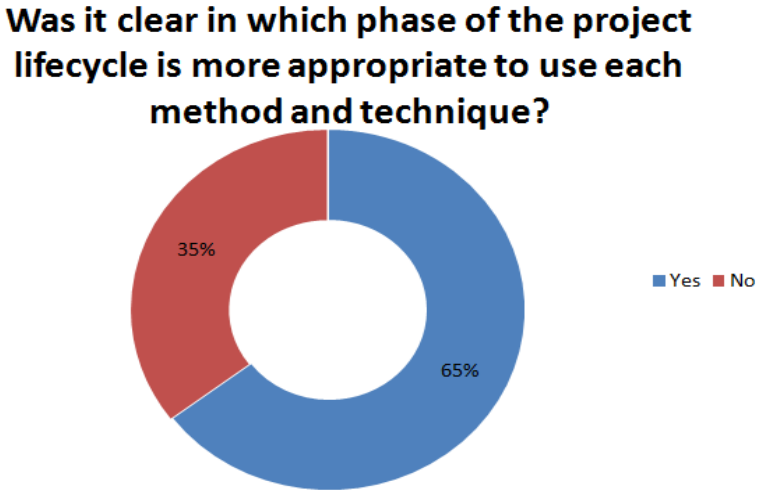

Figure H.18: Question 18

19. If no, why?

Not enough feedback.

The feedback was not sufficient. When making a mistake only the correct answer was given without explaining why.

I could not complete the game

because there was no correct answer if my method/approach got 0 or low points.

Figure H.19: Question 19

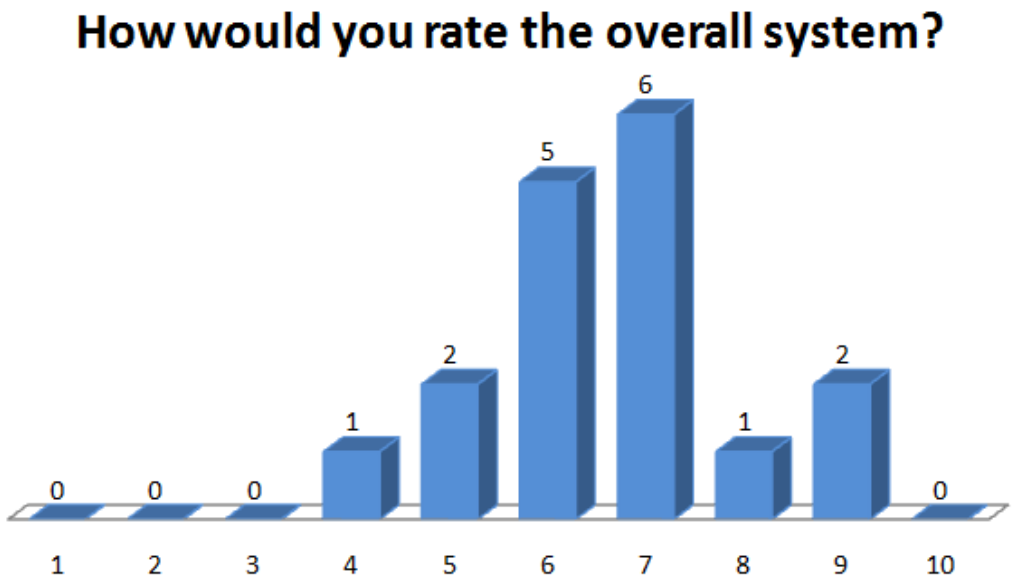

Figure H.20: Question 20

21. Would you like to add any more comments about the overall system?

No comment.

Figure H.21: Question 21

In the interview, only four students took part due to time constraints. During the interview, the author used the same evaluation questionnaire as before (Appendix G). Because the sample was small, it was considered that the creation of graphs would not be representative of their opinion. In particular, 3 out of 4 felt that after the changes, they could find more detailed and helpful information in the help resources, as well as in every step of the game. Two of them thought that they now they could understand better why this number of points corresponded to their selections. Two of them suggested that the user interface could be improved, and only one felt that they still were not clear about the principles of every agile method. Finally, the majority agreed that the new changes improved the overall system.

Appendix I – Interview Questions

During the research phase, some work had to be done to investigate similar systems. One of them was the "Software Management Game" as previously mentioned. The game was implemented by Dr P W Garratt, a lecturer in the University of Southampton. Because of that, it was a great opportunity to meet him and acquire a bit more information about the way that his system works and because of his expertise on the subject, to ask his advice on this project. For this reason, it was considered right to prepare some questions and make this meeting in the form of an interview. These questions were aimed to give the author a bit more understanding of how the "Software Management Game" is structured and to ask his advice on how to proceed with the implementation of a game concerning agile methods (Garratt 1999).

Questions

1. **Who are the users of the game? To which people is it addressed?**
   - Students?
   - Managers?

2. **What is the goal of the game?**
   - To teach traditional project management
   - Entertainment

3. **How is the game structured?**
   - Different levels?
   - Different teams?
   - Is the user part of a team? Or the leader of it?
   - If user not a leader but just a member of the team, how do they take orders?
   - How do players communicate with their supervisors?
   - If user the leader how are their decisions reflected in the system?
   - What hierarchy is presented inside the company?
   - How many people does the team consisted of?
   - Do all the teams have the same project to complete?
   - If the same project, do they still have the same problems during all the phases of the project lifecycle?
   - If different projects, how do you compare the outcomes?
   - How long does a project take to be completed in the game?
   - What resources do the users have? Money, personnel, etc.

4. **How long did your software take to be completed?**

5. **Why did you only implement a game for traditional project management and not proceed with a game on agile project management?**

6. **How do you evaluate the work of each team?** What are the criteria?

7. **How do a team proceed to a different level?** What if the team manages to complete the project successfully, but not as successfully as another team?

8. **How do you ensure that users have learned something from the game?**

9. **If given, how is feedback given to users?**

10. **If a team fails to complete a project, do they start the same project again?** Are they able to see what other teams have done? Do they see where they went wrong? How do you ensure they understand their mistakes and get constructive feedback?

11. **What kind of agile methods do you believe need to be used in a game concerning agile project management?**

12. **If different kinds of methods should be used, how will they be applied?**
    - Different mode for every method? (If the user is the leader)
    - The system itself will pick a different agile method for the teams to represent the difference between each method on a project
    - Application of a combination of agile methods (XP+ Scrum)

13. **What kind of agile methods should be used in the game?**
    - Scrum
    - XP
    - Crystal

14. **If you were doing a game concerning agile project management, what would be the key features of the game?**

15. **What kinds of project does the game has to have in order to have a productive illustration of agile methods in comparison with the traditional project management game?**

16. **What kind of background research is necessary to have a system that well represents the fundamentals of agile project management?**

Appendix J – Project Brief

TITLE
Development of a software management game that helps students to understand agile project management.

PROBLEM
Traditional project management is a very heavyweight approach for small-sized companies. This led to the application of agile methods because they allow iteration during the development process, since the priority of the company is the development of the product and not the documentation. While at university, students that are studying Software Engineering learn both traditional and agile project management. The way that they are taught is vague and it does not give them a clear idea of how agile methods are used in real life.

GOALS
The aim of this project is the creation of a program that helps students to understand the application of agile methods during the development of a project. The program will be a game in which the player will be part of a team that consists of students from the players' course. Their team is required to compete with other teams so they all complete the same project using agile methods. The teams that have successfully completed all the different stages of the software lifecycle applying agile methods, will be able to continue to a more advanced level with a new project. Via this game, the player will be able to learn in more detail about Software Engineering and Project Management in an amusing and interactive way. They will also be able to get a glimpse of how companies work and what tasks they have to perform in order to deliver a new product to the market.

The draft Gantt chart below represents how the project will progress. The first priority will be research on the subject, finding out what has been done on the past on the subject. Then, as soon as research is complete, the design phase of the project begins followed by the implementation of the project. Also, the system will be tested in order to eliminate any faults in it. Finally, the presentation of the viva will take place.

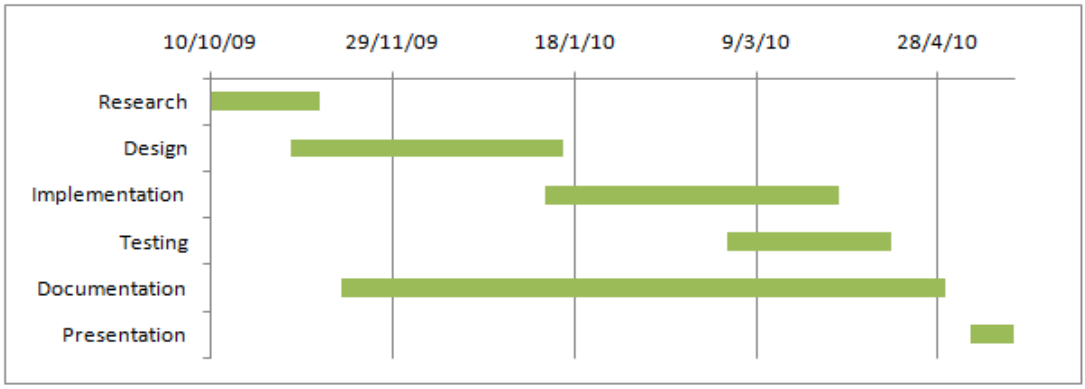

Figure J.1: Draft Gantt chart

Appendix K – Additional Screenshots

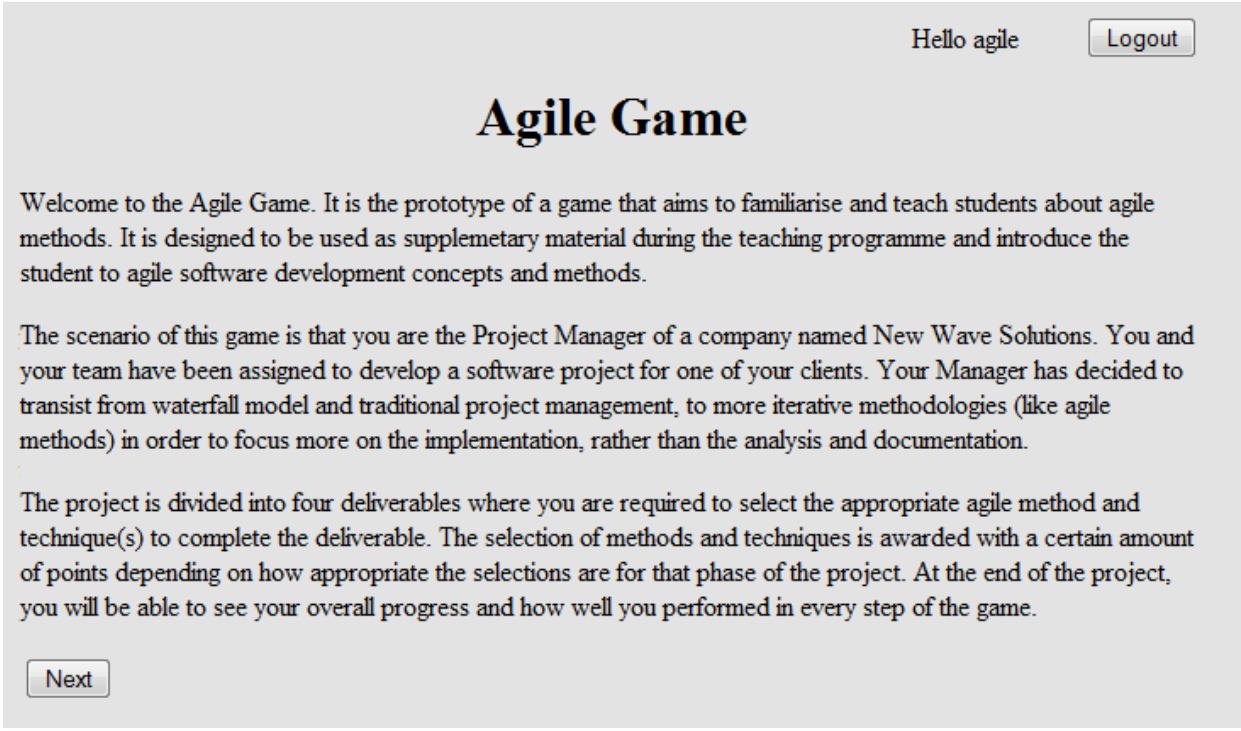

Figure K.1: introduction.php

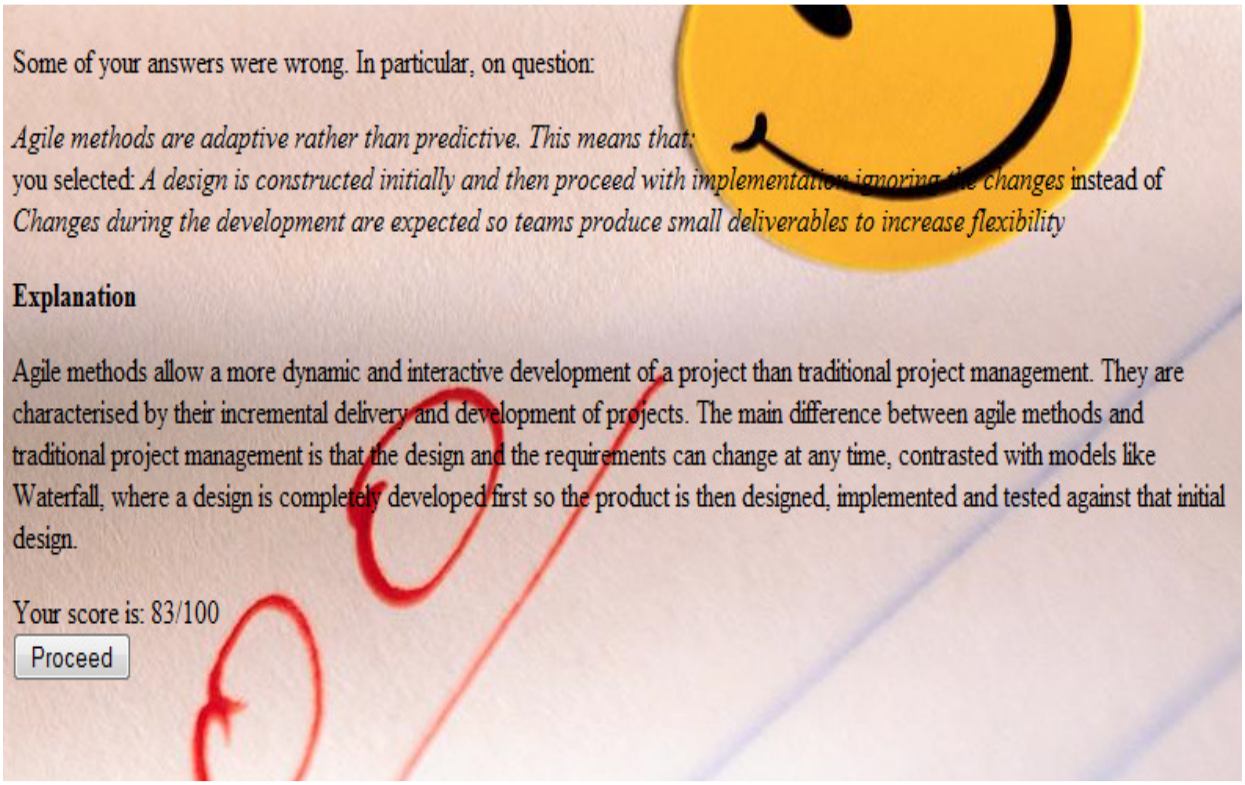

Figure K.2: results.php

Extreme Programming (XP)

XP focuses more on the implementation of the project and not in the documentation and emphasises on the customer involvement and testing. In XP the requirements are expressed as user stories or scenarios from the customer (Figure 1).
This method is a collection of good software engineering practices. When this method is used with other agile methods (like Scrum and DSDM) leads to highly effective development teams that produce high quality software. In general terms, XP follows the notion of doing what it needs to be done and doing it extremely well.
Some of the most common techniques that are used are pair programming and refactoring. In pair programming the development team works in pairs so they develop code more efficiently and with higher quality. They also develop tests before writing the code. Refactoring, allows regular testing and all tests have be successful before new code is integrated into the system.

For more information click here.

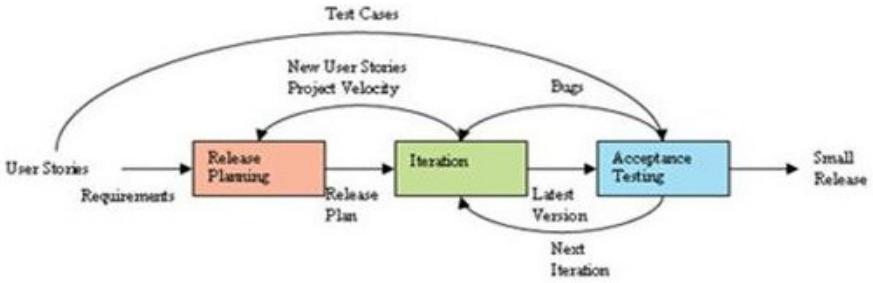

Figure K.3: help.php

Agile Game

Logout

Well done! You have just completed deliverable 3

The method that you used is:

XP Programming

The selected techniques are:

Stakeholder Participation

At this stage, in XP the most appropriate techniques were is Pair Programming because you are in the Implementation phase, as well as refactoring and testing techniques.

The score in this stage is:

1

The points that you have gained overall until this point are:

32

The next phase is deliverable 4 where your team has to create Acceptance Tests. This stage either involves Black box testing to the system which tests the main feature of the system ignoring its inner functionality or user acceptance testing where the client reviews the final system to see if it meets their expectations.

Proceed

Figure K.4: report.php

Agile Game

Thank you for playing the Agile Game

The points that you have gained overall are:

41

Your summary			
Deliverable	**Method**	**Technique**	**Points**
User Stories - Requirements	XP Programming	Code Refactoring	0
User Stories - Requirements	XP Programming	Pair Programming	0
User Stories - Requirements	XP Programming	Continuous Integration	0
User Stories - Requirements	XP Programming	Stakeholder Participation	25
Implementation	XP Programming	Stakeholder Participation	1
Planning- Design	Scrum	Sprint Review Meeting	6
Acceptance Testing	Crystal Clear	Frequent delivery	9

Top 10 Scores	
User	Score
don_carleone	90
dba	75
ama1v07	57
bartsimpson	56
pinkye	49
Hackology	49
mk8v07	47
ben	43
okrums	42
agile	41

In the first questionnaire your score was 17 and in the final questionnaire you scored 57

Finish and complete the survey

Figure K.5: summary.php

Appendix K – CD ROM Index

- Agile Game Implementation
  - Images
    - DSDM.jpg
    - FDD.jpg
    - picture.jpg
    - Scrum.jpg
    - sticker.jpg
    - XP.jpg
  - Files
    - authentication.php
    - connect.php
    - functions.php
    - help.css
    - help.js
    - help.php
    - index.css
    - index.js
    - index.php
    - insert.js
    - insert.php
    - introduction.css
    - introduction.php
    - main.php
    - questionnaire.css
    - questionnaire.php
    - questionnaire.js
    - register.css
    - register.js
    - register.php
    - report.css
    - report.php
    - results.css
    - results.php
    - round.css
    - round.js
    - round.php
    - summary.php
- Documentation
  - iSurvey – Online Question1.pdf
  - iSurvey – Online Question2.pdf
  - iSurvey – Online Question3.pdf
  - Project Brief.pdf
  - Progress Report.pdf
  - Project Description.pdf
  - Final Report.pdf