Software Engineering Group Project
Final Report

Author:	Group 12
Config Ref:	SE_G12_FR_00
Date:	2013-02-14
Version:	1.0
Status:	Release

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB

CONTENTS

1. INTRODUCTION

1.1 Purpose of this Document

The purpose of this document is to clearly state how much we as a group have accomplished on the project. It is also to enable the markers to evaluate our group on how well we have completed the project.

1.2 Scope

This document aims to provide the markers with the concluded summary of how our group project has been. Reflecting on how we worked as a team and how successful we are is shown in the final program and this report.

1.3 Objectives

The objective of this document is to provide the markers with a summary of how the group project has been.

2. MANAGEMENT SUMMARY

2.1 Program

Overall the program we designed, developed, tested and handed in was successful thanks to good communication and a lot of hard work from each member of the team. We decided to split the implementation of the program into the functional requirements, and worked through them one by one. Thus we had things constantly working, with minimal downtime and crashes. For example, some things weren't quite completed until after the functional requirements had all been implemented, like the unregistering. This came along after the first version, and then reading through the test document we generated, we noticed that it was missing, which was not difficult to create to work with the database. Additionally the user can register, then log in which generates a session until the user logs out or the session times when the browser is closed. At one point during the design, we were able to make friends with ourselves on the same account, however this was overcome simply through the friend class method, which additionally allowed us to be able to remove friends and have cross server friends. Alternatively there were some issues with the friends' requirement, which can be found in the maintenance manual. Due to the layout we chose to develop, it allowed for the friends list and monsters list to constantly be on display to the user, thus allowing for quick access to challenge/communicate with friends and monsters. Continuing on from monsters, they also die very quickly. This has been designed to do so, due to proving that the monsters will age, their stats will change in correlation with the requirement we received in the monsters life cycle. As well as dying, monsters should have been injured; however this hasn't been implemented, so instead one monster will always die after battle, whereas the other will carry on living out its life cycle. Due to a monster dying, when a user beings with one monster, this will mean that if the user decided to battle with said monster and was to lose, then they would be out of monsters, thus out of the game, unless the purchased a new one with the money they begin with. To avoid having a possible game over situation, as it is not that type of game, meant we decided to substitute in a new monster, when a user's only monster died. The substitute would be of random value and stature as before. Although we were the only group to have full server to server communication and functionality working, we did use an alternative server slot along with our own to prove that we were able to have full functionality, this including battling, selling and buying, breeding and general friend to friend functionality.

2.2 Documents

2.2.1 Project Plan

The project plan document was successful on the hand in by making certain decisions and creating foundations for moving on in our design. By explaining the intended users and technologies, we were able to decide for ourselves what the best technologies were to use for our group project. We began with initial sketches of the user interface which then turned into the HTML generated design. This allowed us to have a starting point for when we began the implementation of Java and the server to server communication. Additionally, by generating a Gantt chart, we were able to set milestones and goals which allowed us to have good time management and be successful with our deadlines. By having successful Risk Analysis, we were able to predict possible risks to our group, thus allowing us to be prepared for the worst. Due to the Risk Analysis we were able to solve any problems that did arise, with little repercussions.

Alternatively, the Use Case diagram that was generated was a little confusing and poorly laid out. Thanks to feedback and our QA manager we were able to resolve the issue and create a more readable and understandable Use Case diagram.

2.2.2 Test Specification

By generating many tests, which were created with the functional requirements in mind, we were able to look in depth at what was needed and what tests would be needed for these requirements. Whilst looking at the functional requirements we allocated different requirements to different members of our group, thus allowing for us to come together in a meeting and compile with the addition of possible alternatives and additions. With the 11 different requirements we were able as a group to create 69 tests for the program which looked at both client and server side testing with the addition of our JUnit tests. Eventually we had a solid document for the testing however when reading through as a group, we noticed that many of the functional requirements were very similar in terms of the tests we had generated. We decided to merge these into multiple requirement tests, meaning not to repeat the same test for different requirements, since the tests satisfied both.

2.2.3 Design Specification

There is a lot of detail in our Design Specification document, and includes a lot of key information for when we began coding and began to bring everything more together. For example, the Decomposition Descriptions allowed us to look at the different client and server applets, and what each applet will be doing to interact cross server and with each other. Additionally the information of the significant classes, not only give an insight into how the program has been made, but also, what information will need to be included into each class, thus the generation of the Class Skeletons we created to go along side with these pieces of information. The Component diagrams we created allow for front-end use and back-end use and how the different servlets will interact with each other. After we had wrote the Design Document, there was a change in the specification, which removed the gender of monsters and made them asexual. Although this was to make the requirements simpler, it did mean having to go back and remove the inheritance diagram in relation to the gender breeding and fighting. Additionally we had our detailed designs, which allows for more information to be placed into a document, in which when referred back to by ourselves, we can see how to implement certain aspects of the design. To compliment these algorithms, although slightly basic, are a clear, understandable way to implement the code into our program, to allow the battles to take place, the ability for someone to win and of course the breeding algorithm.

However after some feedback, we realized that there was some work to do on our Design Specification document. For example, the UML designs, need to be rewritten to include more detail in the grouped data, additionally the component diagrams weren’t quite laid out correctly, however have been amended so that the servlets are interacting more, rather than just having the singular servlet linking to everything. As stated before, with the requirement change, the female/male details are to be removed, since these are asexual monsters now, which leads to the database having to be edited for the breeding and monster holding data, as well as different breeding and fighting algorithms, with perhaps a bit more detail.

2.2.4 How the team performed

Overall as a group we were quite lucky to have an array of people with different talents. Some were much more technical then others, which in turn helped because they were all gifted in different areas, for example one who was very good with server to server implementation, one who was extremely good at web development, and another who was a very competent java programmer and algorithm developer. Additionally we had a handful of JUnit testers and general testers, and finally some of the people who were not quite as technically gifted as the others, focused on developing the documents to go alongside the program. We had a few issues with attendance at meetings, however the people who did miss the meetings for an array of reasons, had the minutes to read and work to catch up on, which everyone did well. Overall in the development of the program and documentations everyone pulled their weight, although in different ways, which overall led to a successful project.

2.3 Difficulties and their Resolutions

2.3.1 Gantt Chart

The main issue that arose with the Gantt chart was the over complicated design package that we initially chose to use. The package was called Gantt project which we began to generate our Gantt chart in, due to being complicated and non-user friendly, we decided to scrap the package for a more user friendly piece of software. This allowed us to have the ability to quickly glance at the chart and see time frames and our milestones whereas Gantt project did not. Additionally the package we decided to use was more suited to our project needs.

2.3.2 GitHub

We decided to use GitHub to upload and share the work we had done with other group members rather than using email or SVN. However a few issues arose with the use of this software, one of these was the lack of knowledge that some members of our group had with GitHub, this lead to deciding that in one of our group meetings we would have a tutorial session to teach members how to use GitHub. This was successful and allowed the sharing of work to flow, however there were a handful who never quite grasped the concept of GitHub. Additionally the rest of the group had the issue of version control conflicts and merging errors, however this was resolved once correct system file management was implemented.

2.3.3 Inter Group Standards

Our group project has to communicate with other groups projects, taking that into account the standards meeting was arranged in order to develop a set of standards that will make the communication possible. Two people from each group had to attend this meeting and having so many people in one room resulted in having multiple opinions that were preventing standards to be developed in a fast and correct way. Server to server standards were not relevant because of the client change in the specification, additionally some features of the specification were not relevant to the program due to the standards.

2.3.4 Technical Issues

Since we started to develop the program we encountered a number of technical difficulties, for example due to the requirement change meant that some of the code that we wrote was unusable as it catered to an incorrect specification. This then led to database malfunctions which meant that the database we created had to be re-written. Whilst writing the program the main issue we had was the concept of group programming as there were a lot of conflicts in the code and some communication errors between group members. However this was overcome thanks to the decision to have individual group members working on different parts of the program and frequently pushing to our GitHub repository, thus allowing for constantly updated versions.

2.4 Historical Account

2.4.1 Main Project Events

#	Start Date	Event	End Date
TL.1	18/10/2012	Interaction Design	29/10/2012
TL.1.1	18/10/2012	Introduction	29/10/2012
TL.1.2	18/10/2012	Overview	29/10/2012
TL.1.3	18/10/2012	Use Case Diagram	29/10/2012
TL.1.4	18/10/2012	User Interface Design	29/10/2012
TL.1.5	18/10/2012	Gantt Chart	29/10/2012
TL.1.6	18/10/2012	Risk Analysis	29/10/2012
TL.2	29/10/2012	Test Specification	13/11/2012
TL.3	29/10/2012	Design Specification	03/12/2012
TL.4	05/10/2012	First Prototype	10/12/2012
TL.5	10/12/2012	Further Integration	21/01/2013
TL.6	28/01/2013	Integration and testing week	01/02/2013
TL.7	01/02/2013	Delivery of software	01/02/2013
TL.8	01/02/2013	Acceptance Testing	01/02/2013
TL.9	04/02/2013	Document Hand In	18/02/2013

2.4.2 Producing a plan

From the start of this project, we all knew it was going to be difficult and require a lot of planning and allocation of jobs. In the first piece of documentation we generated a Project Plan to help us get the project off to a start. We met 2 to 3 times a week, and then every day during integration and testing week to allow us to constantly come together and see how each other were getting on. As well as meeting each week a few times, the coders of the groups came together a little bit more with the Project Manager to allow them to set some standards within the group, and the coding style. Along with this we had constant communication throughout the project using emails containing the minutes of the meetings if for whatever reason, someone missed the meeting. This then led to one of the group members generating a forum, in which we had the ability to communicate ideas that we had, or possible code that had been generated. Although this was made with good intentions, the use of the forum was extremely minimal, partly due to the decision to use GitHub, everything was stored on there, thus, reducing the need for the forum, it was a good idea, and a possible idea for the future in terms of other projects, however for this one, we felt it didn't meet the requirements we were going to use it for. As well as getting started on creating meeting times, allocating jobs, we had to begin actually generating the plan for the project, which meant looking at Use Case diagrams, sketches by hand that we did, which then led onto a HTML generated design.

2.4.3 Time Management

Managing time is crucial for every project that takes time to complete. Our project was given a number of deadlines that made the time management easier. We as a group were set regular milestones, which didn't just include what was in the Gantt chart, but for example work we had to take home to finish off. Due to having multiple meetings a week, this allowed us to start on a task at the beginning of the week, and having it done by another meeting that same week, meaning that when we were all busy with additional modules, we weren't playing catch up, rather than having a final read through along with the QA manager making sure everything was formatted correctly for the hand in. additionally thanks for having an incredibly detailed task list, we were able to strategize the oncoming issues we had to overcome, and tasks we had to complete, thus meaning that, for example, when we began coding, we knew the order of things that had to be completed, which removed the concept of downtime within the group, most importantly, this allowed every member to be doing something all the time.

2.4.4 Key Moments

Initial Meeting

On the initial meeting we quickly worked out everyone's strengths and weaknesses, and by the second meeting had formally chosen people for the main roles. Llion became the Project Leader, Yarrow the deputy Project Leader, James the Quality Assurance Manager, and Sindre in charge of Standards for the project. We also settled into our group quite quickly - there was a healthy rapport between the Leader, Deputy Leader and QA, and the other members of the group followed suit.

Yarrow and Sindre, given the chance to choose which version control system to use for the project, put forward that the group should chose Git. This was for two important reasons - one, prior knowledge, which made teaching the rest of the group easier, and two, so that we could maintain the repository outside of the university network.

Initial interaction design

We quickly got started on this task, splitting it up effectively - Sindre handled research on Glassfish while Yarrow wrote about the pros of Git vs SVN. The whole group agreed on the screen designs for the web front end, which were drawn up in one meeting. The use case diagram was constructed by Dan, and Pavel made the risk assessment. Llion and Yarrow combined their efforts to make a Gantt Chart, initially with Gantt Project, which turned out to be too complicated, so a simpler chart was made.

Due to high motivation and good division of tasks amongst the team, the document got finished a week before the deadline, giving us the time to check over the document and review it before handing it in.

Testing plan

As we had time before the hand-in of the Initial Interaction Design, we looked over and split up the tasks for the Testing Plan by the functional requirements in the design specification. Where there were linked functional requirements, we grouped them and gave them to one person, for ease of writing. Everyone went off and wrote up test tables that had tests which target the boundaries of the functional requirements e.g. entering bad emails, and that test that the pages are linking correctly.

Similarly to the last document, we finished the document and reviewed it a week before the deadline.

Full design and prototype

For the class diagram, we spent several meetings together as a group agreeing on what the diagram should be like, with the less coding-inclined contributing their ideas as well as the coders. Stan and Sindre took charge of the construction of the diagram. As a group, we discussed the significant algorithms and Filip took over that task.

The prototype was built by Stan, Sindre and Filip, and at the demo presentation we were able to show that our application could log in, create accounts and view their details.

2.5 Final State of Project

Throughout the project we encountered many problems which led to certain parts of the program not working, or documents not quite being correct. Due to design and planning however we were able to look back into our previous documentation, and figure out what was going wrong. In the final version of our program, we were able to have server to server working, except for, the remote users not being stored in the database, thus the user information is needed to be retrieved from the remote server when it finally is needed. This makes the site slow since it has to wait for a remote response. Although the site is slow, there is an easy fix for this, in which ideally we would of liked to have implemented (details of this can be found in the Project Maintenance Manual).

Additionally there was the server issue of a 500 error which was caused when the remote server gives a bad response, thus causing some of the pages to redirect. (Details of this can be found in the Project Maintenance Manual). As well as the server to server communication being successfully implemented, so was the cross server; friends, breeding, buying/selling and fighting. FR6 was successful, although we had certain issues when unregistering not updating the database; this was resolved before the final hand in. all other aspects of this functional requirement are all working well. Additionally, functional requirement 7, the start-up of the software boots fine in all of the University browsers, and has an option to log in and register, thus storing them in the database and allowing for the retrieval of data to login, in the future. The final version does include a logout option, which will terminate the current session and bring the user back to the log in screen. As for FR8, this linked in closing with the FR6 and was mainly the requirement for the actually display rather than the interaction. After a user has logged in the list of their monsters and friends are displayed on the sides of the window, this remains on all the pages of the website, when a user clicks on one of their monsters a display is shown of the stats and status of that monster. Coinciding with FR9 when a friend sends a friend request to a user a notification of this is shown in the centre of the screen and on the Friends List side bar a new entry is shown. The user can then accept or decline the battle request by clicking on this message.
A user can send a friend request to another user by typing the email address of the user in the search bar at the bottom of the Friends List side bar, this will send a notification to the player. Friend requests to a user are displayed as in the Friends List side bar, a user can then accept or deny these requests by clicking on the request entry. Alternatively some issues that arose with the adding friends included non-registered users, input email field, no notification and rejecting a friend request sending the wrong information (details of this can be found in the Project Maintenance Manual).
Users can conduct fights and the loser’s monster is removed from their monster list, both players receive a notification detailing the battle and the winner receives the prize money, this conforms to FR10.
There is a page accessible from the Main page that shows a list of all a players’ friends and orders then by their wealth, and so FR11 is also fulfilled.

2.6 Performance of each team member

2.6.1 Stanislaw Klimaszewski – Lead Designer

Stan was very enthusiastic from the start, taking on the role of "Lead Designer" where he would stand in front of the group during the design process, sketching up both the user interface at first and later the class diagrams and database design. Stan was good at encouraging the group to discuss the project and how we could approach the implementation of our Monster Mash game. Stan's attendance was outstanding and he always contributed during group meetings. Stan also created the first HTML "mock-up" of the interface with Filip and then moved on to building the first semi-functional prototype with the java database.

During the implementation process Stan's technical abilities were superb, and despite some initial problems with communication with other parts of the development team, Stan implemented many of the games functions while working well alongside his team members.

2.6.2 Sindre Knudsen Smistad – Standards Officer

During the initial steps of beginning the project Sindre took on the role of "Standards Officer" where he would attend bi-weekly standards meetings where he and other groups would discuss how server to server integration would work between groups and subsequently the creation of the API. Following these ""standards meetings" Sindre would also report back to the rest of our group regarding the progress of any standards and any changes to the specification. It was apparent that Sindre had taken notes during the standards meetings as his reports back always aided in the design process and if our group disagreed with some of the proposed standards Sindre would bring them up in the "Standards meetings".

When it came time to implement our designs, Sindre worked mainly on the server to server integration. Sindre's work on the API meant that other members of the development could continue their work and later implement functions to work with another server. Sindre's attendance was brilliant and he always showed up on time, ready to work.

2.6.3 James Upshall - QA Manager

James took the role of QA Manager, it was his responsibility to assure the quality of everything the group produced, whether it be the documentation or the quality of the code in the final release version of our program. From the beginning James was also in charge of taking the minutes of the meetings, he would circulate these throughout the group which helped the group take actions and prepare in future meetings. James rarely missed a meeting and always played a key role in the meetings by contributing ideas and suggesting what needed to be done.

During integration testing week James worked hard on many different areas of the project, he wrote some JUnit tests, implemented the breeding algorithm and made sure that the code being produced by all members met the "Java coding standards".

2.6.4 Yarrow Paddon-Butler – Deputy Leader

Yarrow took on the role of deputy project leader, her attendance was brilliant and she was always ready to work. Having plenty of experience with "Git" meant that Yarrow was considered our version control expert; initially she helped the people who hadn't used "Git" before to set it up correctly by providing a very helpful tutorial. Her experience with Git also helped during the implementation process when the development team came across conflict problems. Yarrow also initially helped me with the Gantt chart, but we later decided against using the software package we used at first.

Yarrow's experience with Git proved very usefully during implementation week as the group initially had teething problems while all working on the project at the same time. Yarrow was an asset to the group who was always willing to research anything the group didn't fully understand and report back.

2.6.5 Filip Zajac – Coder

Eager to begin the implementation, Filip was always active in group meetings discussing his ideas and how we could implement some of the more complicated algorithms; he was also responsible for writing the pseudo code for these during the design process and later produced the test tables for testing them. Filip also worked on creating the HTML interface with Stan and the first working prototype.

During programming Filip worked hard on implementing some of the more sophisticated algorithms, such as the "Monster Mash" fighting.

2.6.6 Daniel Hopkins – QA Assistant

During the design stages Dan worked on the use case diagrams which helped the group break down the project and its required functional requirements. Dan actively participated in discussions when present in group meetings and acted as "Deputy QA Manager" by creating the outlines for documents and identifying what needed to be done.

During integration and testing week Dan wrote some of the jUnit tests and implemented out high level tests that were planned in the test specification. During carrying out these tests Dan found some obscure bugs that may otherwise have been missed. After completing the majority of the tests, Dan was left in charge of beginning work on the "Final Report", he worked extremely hard on completing the parts he could and notified other group members on the parts he required them to complete.

2.6.7 Pavels Lode – Chief Tester

Pavels attended meetings and despite being rather quiet to begin his confidence and contributions to the group developed as the project went along. For the first document Pavels worked on analysing the risks our projects may face and their possible preventions and solutions.

As integration and testing week started Pavels wrote many JUnit test for the code the development team were producing. His tests proved useful in discovering bugs that could be ironed out early in the implementation process. Pavels also worked with Dan on beginning the documentation required for the final report.

2.6.8 Llion Vaughan – Project Leader

Llion was the Group leader, it was job to organise the group by scheduling and delegating the work efficiently. During the early stages of the project, Llion alongside Yarrow produced a Gantt chart which was used for establishing milestones and estimating how long each task will take. Llion also created a timesheet system to track of the hours that group members were putting in. Llion was also in charge of organising group meetings and always attended them.

Before integration and testing week, after a meeting with group members Llion produced a schedule for when work should be carried out and a task list of all the things needed to be done. Throughout the week Llion wrote emails at the day's end to all group members with a summary of the current day's work and a task list for the next day. Llion delegated the work well; making sure everyone had something productive to do. Llion also wrote some JUnit tests and the basic algorithm for monster name generation.

2.7 Critical Evaluation of the team and the Project

The group performed very well, with no conflicts and everyone having their specific roles within the group. We were very lucky to have a group with members of diverse skillsets and experience. The time spent in the first few meetings “getting to know” the group members, their skills and preferred area of work meant the group leader could delegate tasks efficiently.

The group settled into working as team very quickly and everyone got along well, which meant that communication within the group was great and everybody’s ideas and problems were expressed and listened too. At times not everyone was present at meetings, usually with good reasons, but this did not mean that anyone avoided being delegated work. When group members were delegated work, they completed it on time to the best of their abilities and expressed any problems they had.

During integration and testing week the group worked 9am-6pm, meaning everyone worked at least 9 hours every day. During the implementation process there was a slight communication problem between Stan and Sindre, as Stan was implementing functions that worked locally he failed to communicate with Sindre who was developing server-server integration, this caused some delay as code needed to be rewritten to function locally and with another server. Stan, Sindre and the whole group learnt from this communication lapse and the group leader made sure that members communicated their progress and what they were working on regularly.
A possible improvement to the project would be to complete more implementation work over the Winter break, so more time in integration and testing week could have been spent on perfecting our final product and further testing with another server. It was also a shame that there wasn’t another group to fully test our server-to-server integration with after working so hard on getting it implemented.

Everybody in the group learnt many lessons throughout this process; it was most people’s first time working on such a large project with so many group members. It was also an experience much like working for a real software development company, with everyone having their roles and responsibilities. Integration and testing week gave everyone a taste of working real office hours.

REFERENCES

DOCUMENT HISTORY

Version	*CCF No.*	*Date*	*Changes made to document*	*Changed by*
1.0	N/A	2013/02/15	Implementation of all major aspects	G12