Software Engineering

Project Plan

Group 11

Group 11 - Project Plan

Table of Contents

1. Project introduction.

Project outline.

This project is a development for a game designer who requires the implementation of a competitive two-player strategy game. Both players provide an ant-brain (a simple finite-state machine). To run the game, the brains are uploaded into an ant world with two ant colonies, one for each player. The ant world will then simulate the behaviour of both ant colonies, using the ant-brains supplied by the players. The match is won by the species with the most food in its anthill at the end of 300,000 rounds

The program should consist of a tool to check that the world and ant brains being used by the game are legal according to the specification given, before running the simulation using the given ant brains and ant world.
We will enable all of these elements to be merged into a single program and compute results and statistics as to which of the colonies was most efficient. The architecture of the game process itself, how it should be played and built is entirely up to the software engineering team. For instance, decisions can be made surrounding:

- The level of visualisation – text, 2D, 3D, colours
- Software type – applet, full application
- Playable machine – console, PC
- Programming Language
- Gaming features – scoring, tournaments, statistics, high-scores

Our software house, composed of eight engineers is divided into four teams:

- Analysis Team;
- Design Team;
- Programming Team; and
- Quality-Assurance Team

Each team will have a specific set of tasks and subtasks assigned. As would be expected in a commercial environment, our work will not only be restricted to programming, but will reflect the full process of an efficient software engineering, including the planning, analysis, design, documentation and testing of the final product, committing to organisation, procedures and standards described in a quality manual.

Project schedule.

The project schedule is split up into five phases:

- Analysis;
- Design;
- Programming
- Testing; and
- An epilogue, which will compile the user documentation and present the project to the customer.

In the requirements stage, we take the customer specification and break it down into a workable blueprint for the project by using several methods. This is the stage in which we detail our goals and objectives of the project. This stage will also be used as a springboard for the design team, who will take this document and use it to create a more robust design. The aim is to have this completed by March 12th, 2012.

The design stage takes the requirements analysis and fleshes it out into something that can be easily read and used by the programming team. This involves taking the original ideas set out in the requirements analysis and creating a more advanced idea of how the system will work. As this stage progresses, the design becomes increasingly complex and detailed, providing us with an accurate way of describing how the project will be in its final stage. As this is split up into 2 phases, we are aiming to complete the higher-level design by 12th of March 2012, and the detailed design by 1st May 2012.

The programming stage produces a working model of the game, implemented according to the specification given by the design team. The decision on the technical implementation rests with the programmers, so long as the final working model meets all of the requirements.

The testing stage is the final stage of the project. This is where we repeatedly test the systems functionality and corrects bugs in the code that cause problems. Both this, and the programming phase will be accomplished by 11th June 2012.

To end the project, we finalise the user documentation and present the completed product to the customer.

2. Conflict resolution plan.

Potential conflicts and resolutions.

- ***Withdrawal from the team by a team member.***
  In this case, a meeting will be held between the remaining team members, with the focus around which individual is most suited to take on the extra work that has been left by the departing team member. This will be dependent upon the suitability and competency of each individual in the role.
  Preferably, the person taking on the role will have done so voluntarily. If not, then a vote will be passed to decide who will take on the role. If no one gains a majority vote, then the project manager will make a final decision.

- ***Disagreement between two members over a project matter.***
  If this occurs, then those in disagreement need to present the issue to the project manager initially. If it's a minor decision then the project manager will decide. If a bigger problem, such a design decision or something that will affect the work of another team, then the project manager will present the problem to the other members at the next weekly meeting. A vote will be passed to reach a decision, and if no conclusive result comes from the vote, then the project manager will make a final decision.

- ***Personal issue between two or more members of the team***
  If this occurs, then the problem needs to be raised with the project manager. This will remain confidential, and the project manager will act as mediator and resolve the issue. It is important those involved act in a professional manner, and slandering or discussion of the issue with other team members is not acceptable. If it is impossible for the team members to work together, then the issue will be raised with Martin Berger, and any further decisions made by him.

- ***Personal issue between the project manager and another member of the team.***
  If this occurs, another member of the team will act as mediator whilst the problem is resolved. It is vital that during this process that the team member does not ignore the project manager's instructions on the work, and equally that the project manager does not discriminate against the other team member.

- ***Repeated absence from meetings by a team member.***
  In the case of a team member missing two consecutive meetings without telling any other team member beforehand, then the project manager will get in contact asking for an explanation. If this happens repeatedly, then the project manager will notify Martin Berger of the repeated absence, and this will be taken into consideration in the peer assessment.

- ***Work from a member judged too poor compared to requirements.***
  Should a member not withdraw but still provide unsatisfactory work according to the team's expectations and standards, there needs to be a plans to resolve this before it negatively impacting on the whole groups mark. This would initially be discussed between the manager and the person involved, where a possible agreement can be made. If this is not possible, then the issue will be presented to the group. If the cause of the unsatisfactory work is due to a personal problem, then the project manager will take it into account. However, if it is down to unwillingness of an individual to put in the required effort, then this will have the same ramifications as the repeated absence and it will be taken into account at the time of peer assessment. If it is just that the member in question doesn't have the necessary skills to efficiently accomplish the tasks he/she has been assigned, then it needs to be raised by that individual. In that case one or more other members will be chosen to help depending on available free time and similarity of tasks in the project. If a meeting is not held due to lack of involvement then the decision will be discussed by the group as per point 1

- ***Conflicts arising from peer assessment.***
  The marks given in the peer assessment review should be based solely on the mark scheme set out in the peer assessment plan, and should not take into account any personal issues one may have with other team members. If a team member considers that their mark does not reflect their contribution, then they should in first instance raise this with the project manager. The project manager will then mediate between the markers and the individual and those who feel they have been unfairly marked.
  Should the project manager be the one who they have cause to be reassessed, then someone else in the group will be appointed to act as mediator.
  It is important to not that the final decision to change any marks lies solely with the marker. The mediator can only question the justification of marks and recommend any change if necessary.

3. Phase plan.

Project phases.

- ***Analysis.***

1.1	Analysis modelling	
1.2	Requirements specification document	*milestone Monday 12 March, Spring Week 10*
1.3	Acceptance criteria document	*milestone Monday 12 March, Spring Week 10*

- ***Design.***

2.1	High level design	
2.2	Detailed design	
2.3	High level design document	*milestone Monday 12 March, Spring Week 10*
2.4	Detailed design document	*milestone Tuesday 1 May, Summer Week 3*

- ***Programming.***

1.1	Coding	
1.1.1	Preliminary coding	
1.1.2	Ant simulation	
1.1.3	Visualisation	
1.1.4	GUI	
1.1.5	Design ant brain	
1.2	Refactoring	
1.3	Integration	
1.4	Optimisation	
1.5	Deliver source code	*milestone Monday 11 June, Summer Week 9*

- ***Testing.***

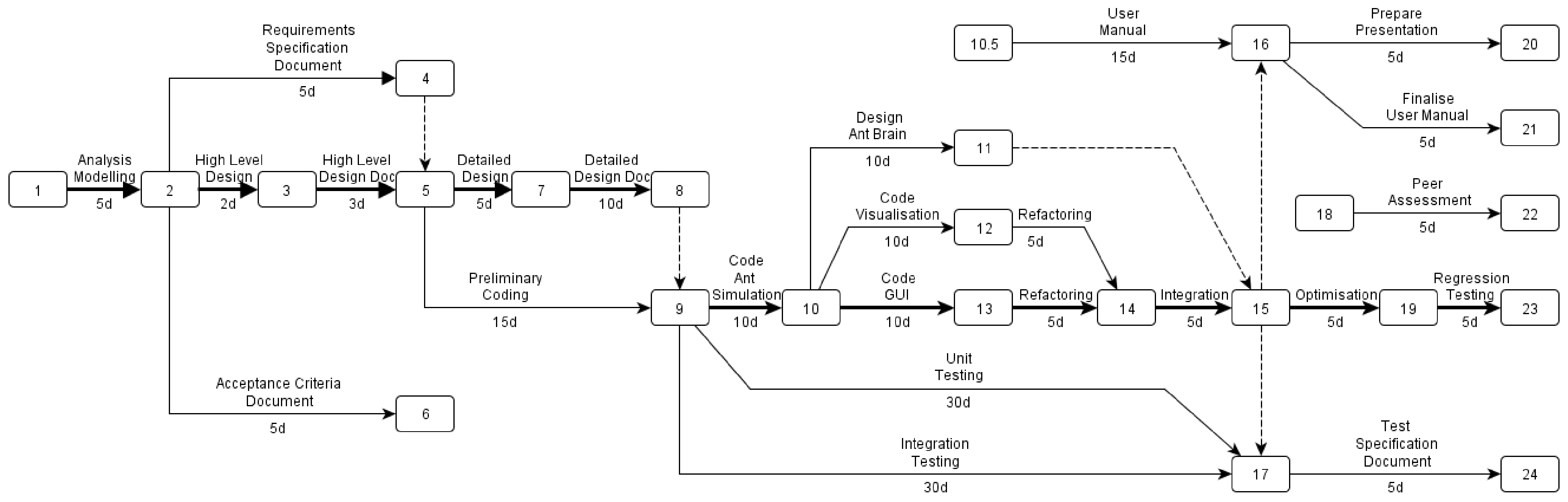
4.1	Unit testing	
4.2	Integration testing	
4.3	Regression testing	
4.4	Test specification document	*milestone Monday 11 June, Summer Week 9*

- ***Presentation.***

5.1	User Documentation	*milestone Monday 11 June, Summer Week 9*
5.2	Peer assessment	*milestone Monday 11 June, Summer Week 9*
5.3	Prepare presentation	*milestone Wednesday 13 June, Summer Week 9*

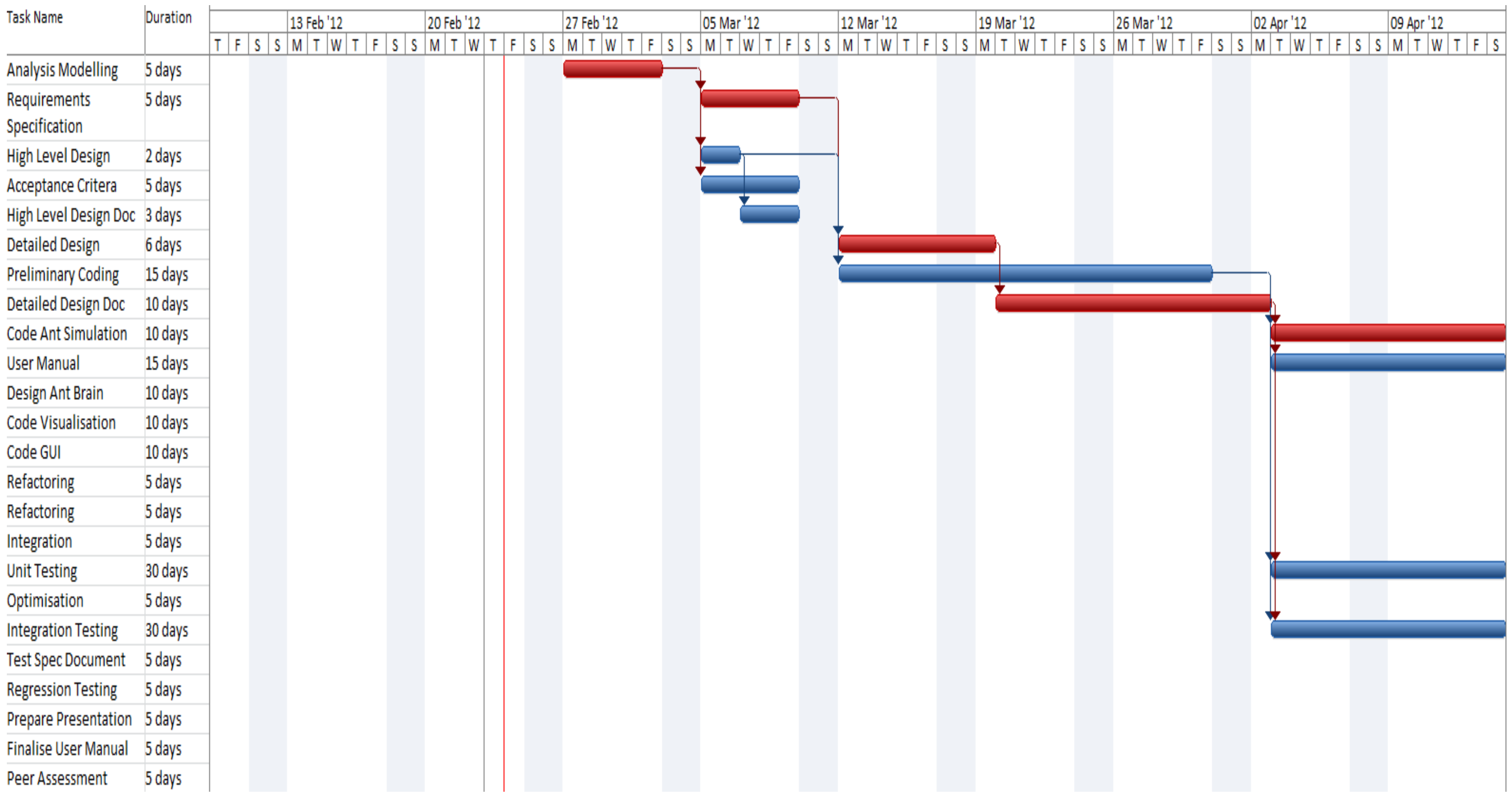
PERT Network.

→	Activity
⇢	Dependency without a resource
➔	Critical path



Gant Chart.

The Gant Chart visuals the project schedule by dates.



Task Name	Duration
Analysis Modelling	5 days
Requirements Specification	5 days
High Level Design	2 days
Acceptance Critera	5 days
High Level Design Doc	3 days
Detailed Design	6 days
Preliminary Coding	15 days
Detailed Design Doc	10 days
Code Ant Simulation	10 days
User Manual	15 days
Design Ant Brain	10 days
Code Visualisation	10 days
Code GUI	10 days
Refactoring	5 days
Refactoring	5 days
Integration	5 days
Unit Testing	30 days
Optimisation	5 days
Integration Testing	30 days
Test Spec Document	5 days
Regression Testing	5 days
Prepare Presentation	5 days
Finalise User Manual	5 days
Peer Assessment	5 days

4. Organisation plan.

Staff organisation.

- ***Team Members.***

Andreas Nicholaou	Analysis Team & Quality-Assurance Team
Mark Purser	Analysis Team & Quality-Assurance Team
Eleanor Shakeshaft	Analysis Team & Quality-Assurance Team
Ben Watt	Analysis Team & Quality-Assurance Team
Oliver McCarthy	Design Team
Mark Merriman	Design Team
Robert Johnson	Programming Team & Project Leader
Joanne Robert	Programming Team

Note: The Analysis Team and the Quality-Assurance Team have been merged to relieve pressure on these teams during the exam period.

Configuration Management

- ***Version Control.***

For version control, GitHub will be used (project available at https://github.com/Rob-Johnson/Software-Engineering-2011-Group-11). This tool allows us to upload, read and adapt files on our own dedicated file space, as well as being able to leave comments on the changes that have taken place. By doing this, both small and large scale changes can be logged.

We will also be using DropBox to keep master copies of project documents. By doing this we have another way to back-up our files and it provides us with a more intuitive outlet for our final documents in the website.

- ***Document Outline.***

When documents are uploaded, they will be given a title and a version number in the format shown below:

Document Name: Version

GitHub tracks dates and the names of the authors, additional information in the file name are seen as superfluous at this point.

- ***Change Management***

Preliminary documents will be kept both within the workspace of the team member, but also online as well. By doing this, quality assurance on documentation can be performed by any team member that wishes to do so. This also allows us to track a way a document changes from its inception.

Committed documents will be reviewed after each commit to GitHub. When a project its being modified we will have the last audit date after its last modification date. The last modified version document will be formed as 0.1, 0.2 etc.

Checked out documents are the ones that have being checked in, but are passed out to a member for a checking? When a document is being edited is kept on the workspace of the team member and after the user finishes editing it has to be checked in with version number ending with a letter such as 0.1a or 0.2b. Once the document has been renewed it can be checked in.

- ***Checking Files in and Out***

GitHub will store the files electronically, although individual members of the group will also store a backup version of the project.

- ***Project Logs***
  GitHub tracks changes in a documents lifetime, including the username of the person who lasted altered it as well as comments they leave to summarise change that have been made. As a result, this will be the way we will be logging our project.

- ***Memos***
  Discussions and memos will take place on the Software Engineering Forum, which is automatically archived.

Analysis team responsibilities.

- ***Overall Responsibilities:***
  The Analysis team must interrogate the customer's specifications and requirements, scrutinise the data and present the principal facts and tasks that the software engineering team must adhere to when developing and producing the game. This will be the basis for the creation of the software design and the set of requirements that will be validated when the software is done.
  During the development the team will take meeting minutes, make sure documents are done appropriately and produce a team website where they will upload project deliverables.
  Once the program is finished they must present the customer with documentation which will explain how they can install and use the program, within the licensing agreement.

- ***Project Phases:***
  The project website will store all the deliverables submitted during the project and will be completed by 1st March 2012. It will contain a list of group members, and any further information deemed necessary to keep on the site. It will be designed by Eleanor Shakeshaft and Andreas Nicolaou and is hosted at http://dl.dropbox.com/u/12957105/Website/home.html

  Analysis modelling is the key component, which needs to be completed before the design team can start. It will hold all the customer requirements and the requirements specification, which itself will specify details from the goals of the game or the objectives to the security requirements.

  Finally the User Documentation will be finalised near the end of the project, this will reflect the product delivered, not the envisaged product designed at the beginning. It will comprise of 3 sections, which need to be accessible via the team, website, they are: the licensing agreement, installation guide and user manual. The installation guide is aimed at system admins who will install the software. The user manual is aimed at the users of the software.

Design team responsibilities.

- ***Overall Responsibilities:***
  The Design team's responsibilities centre on the structure of the project by taking the information given by the analysis team and fleshing it out into a full design. This is split into two stages: the high-level design and the detailed level design.

- ***Project Phases***
  The high-level design involves taking the requirements specification and implementing it. The central part of this section is taking the classes and objects identified by the analysis team and expanding or adding to them, ultimately creating an architectural design, which will be detailed with various diagrams in UML notation. This is also the stage where we decide on the coding style of the project, which will remain in place for the projects lifetime. Opportunities for concurrency are also identified, as well as local mechanisms within the system and the policies on how to handle them. Finally, a requirements cross-reference will be created to bridge the gap in-between the analysis model and the design model, detailing how the architecture has evolved from its inception.

  The detailed level design is about taking the high-level design and detailing every aspect of it, taking the original architectural design and describing it in even more detail with UML diagrams. This includes detailed information on classes, object states, methods and the identification of abstractions and opportunities for inheritance. Changes that have been made since the high-level design will be identified and justified. The purpose of this part of the design is to provide a blueprint, which will allow the programming team to implement it quickly and efficiently.

The design team consists of Oliver McCarthy and Mark Merriman, with the duties split into different areas, which will allow us to build the design separately before meshing them together as a final design.

Programming team responsibilities.

- ***Overall Responsibilities:***
  The programming team are responsible for implementing producing the end software in line with the requirements of functionality and design specified by the design team.
  It is important that although the programming team has a freedom over how the final functionality and design of the software is implemented, it meets the requirements set out by the design team as accurately and efficiently as possible.

- ***Project Phases***
  There are numerous phases to the implementation of the design, which can be split into either high or low level models. The low level design is often dynamic and can change, so it is hard to predict the exact phases. However, the high level programming can be viewed as followed:
  - Syntax Analyser for Ant Brain;
  - Syntax Analyser for Ant World;
  - Ant World generator;
  - Game simulation including server-client functionality;
  - Game visualisation;
  - Game GUI; and
  - Ant Brain for use in competition;

- **Technologies**
  There are a number of technologies that can be used by the programming team so that the product is easier to implement and manage.
  - The project will be programmed in Java. This was chosen because of the numerous packages and libraries available to ease the implement the required features. For instance, Processing will be used to implement the simulation, too take advantage of the graphical tools it offers.
    Furthermore, the portability of Java is important, allows it to be easily run across a most platforms. This will be of great advantage for the company wanting the game, as does not unnecessarily decrease its target audience.
  - Common IDEs will be used to develop the game.
  - Version control tools such as Git make the game easier to manage and log changes too. It also allows multiple users to be able to work on the program at the same time, as it intuitively combines the changes made by different users.

Quality Assurance team responsibilities.

- ***Overall Responsibilities:***
  Quality Assurance will contribute a vital component to the project ensuring that all documents, code and procedures meet a suitable standard. To meet these ends, third-party tools will be used where necessary including bug tracking, continuous integration and testing framework. Another essential component of QA is the correct use of change control. To this end, the QA team will ensure that the selected change control system, Git, is maintained and used by all team members for all documents and code. The QA team will also ensure branching is used and set up development, QA and release branches for controlling release of working software.

  The QA team will specify how code should be tested, and write unit tests as the program is developed. Subsequently, the QA team will perform integration testing, as the program gets larger, and finally validation and regression testing to ensure that the program meets the specifications specified during the analysis phase.

Like the other teams, the Quality Assurance team is responsible for their parts of the phase and organisation plans. The QA team is also responsible for ensuring that the other teams work to the guidelines given in the Quality Manual

The QA team will take the minutes of the deliverable reviews.

Project Phases

The initial goal of the QA team is to deliver the Acceptance Criteria Document. This key document defines the over-arching approach to testing that will progress throughout the project life span. The Test Environment is described which defines a benchmark of conditions and recourses that must be met to run tests. The black-box Acceptance Tests are also defined. These tests are linked to and dependent on the analysis modelling that is completed by the Analysis Team.

Most of the testing phase occurs concurrently to the work performed by the Programming team. Unit testing and integration testing are the two essential types of test here. The JUnit testing framework will be used with code injected in to the functions created by the programmers. The unit tests are automated and the code is kept in a constant state of passing the tests. Similarly, integration testing will be a fluid process that ensures that the code is continuously integrated and tested.

The Test Specification is the final document created by the QA team. The results from all the testing are collated in this document. All procedures that the QA team have used are described

5. Peer assessment plan.

Each person will have 100 marks to allocate between the 8 members of the software team, if everyone was to be marked evenly then each member would get 12.5 marks. However, each person must consider how each member has worked over the project in the following areas:

- *Relating positively to other group members: allowing other members to have a fair chance to contribute, responding constructively to each other's contributions.*
- *Being a teacher: explaining things to others, helping other group members to learn and grow.*
- *Attendance and participation out-of-class.*
- *Quality of work.*
- *Overall level of participation.*
- *Suggesting ideas.*
- *Performing routine tasks.*
- *Keeping the group going when things are rough.*
- *Sorting out problems.*
- *Initiative: Generating ideas for the activities and methods of solution, Findings ideas from other sources.*
- *Commitment: doing a fair share of the work, meeting deadlines, attending meetings and being punctual.*

If members drop below minimum expectations (meeting deadlines, overall good quality of work, commitment and attendance) then they should get less than the average mark of 12.5, however if they do more than is required due to excellent skills (and sometimes as a result of other member's lacking in areas) or being an exceptional teacher to others, suggesting good ideas and keeping the momentum going then they should get over 12.5. However, the first bullet point needs to be considered also, if a member takes over in the sole hope of getting more marks and prohibits some members of reaching their full potential then this need to be considered also.
As the total marks are 100, then the allocation of them needs to be measured carefully – giving someone (including yourself) a higher mark will result in needing to give the others a lower one.

The marks will be tallied by the Project Manager and will be calculated in the following way:

1. The individual peer mark (average mark given) is calculated for each person, excluding the mark they gave themselves – in grey. (IPM)
2. The average peer grade is calculated for the group. (APG)
3. The group mark is the average grade from all deliverables. (GM)
4. The final mark is calculated using the formula: GM+ 2 * (IPM - APG)

An example of how this would work is shown below:

	Person 1	Person 2	Person 3	Person 4	Person 5	Person 6	Person 7	Person 8	Total Marks
Person 1	24	4	14	13	15	16	2	12	100
Person 2	9	35	10	9	14	13	1	9	100
Person 3	16	6	19	12	16	17	3	11	100
Person 4	18	5	13	16	18	15	2	13	100
Person 5	23	5	12	12	17	14	3	14	100
Person 6	26	6	13	12	15	15	1	12	100
Person 7	0	0	0	0	0	0	100	0	100
Person 8	21	4	14	12	14	13	2	20	100

									Average Peer Grade
Individual peer mark	16.14	4.29	10.9	10	13.14	12.6	2	10.14	9.9

Group Mark Example:	70							
Final Mark Example:	82.5	58.8	71.9	70.2	76.5	75.4	54.2	70.5

All group members will agree on and sign off the final peer assessment; if any member disagrees with their marks, please refer to the conflict resolution plan.