Project Plan

Version 1.0
February 10, 2006

CPSC 319
Team 7

UBC Botanical Garden and Centre for Plan Research

Dr. Andrew Riseman

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Jack Liu	This is the initial draft of the Project Plan.	Feb 05, 2006

Table of Contents

1. Introduction

1.1 Overview

This document will outline the plan for the development of "XGene 360".

XGene 360 is an application that will simulate plant breeding between diploid plants of the same species under similar environment conditions, employing basic concepts and principles of plant breeding and quantitative genetics. The application will be written as a PHP-based web application, which will be accessible by any web browser that supports HTML 4.0 and CSS 1.0.

The main goal of this is project is to:

- Help students further understand basic concepts and principles in quantitative genetics
- Allow professors to interact with students through problems and feedback
- Allow students to manipulate plant genetics through the process of reproduction
- Provide students with a simulated reproductive environment

1.2 Deliverables

The estimated date of deliverable is the beginning of April; however, the exact date is to be determined.

Upon completion of this project, the deliverable will include:

- Documentation – instructions for installing the application and user documentation
- Application Files – the necessary files for the application
- Installation Files – the necessary files to help user installing the application

1.3 Assumptions and Constraints

1.3.1 Constraints

- The project must be completed within 13 weeks.
- E-mail and IM are the primary forms of communication among the group.
- Communication with the client must be minimal; questions must be aggregated and asked only by the communications manager.
- The development of all documents and products will be produced by individual, and followed by the group or peer review. The individual will take the feedback and refine the document or product.
- All decisions will be rationalized and recorded.

1.3.2 Assumptions

- Research will be reviewed before all meetings.
- Group meetings will take place from 2:00pm to 3:00pm on Tuesdays and Sundays.
- All work will be completed on time and be moderated by the phase leader.
- Once decisions have been made, no new discussion will arise unless new information is made available.
- In the event of disagreement, the project manager and the phase leaders will make the final decision; if the project manager and the phase leaders cannot reach a decision, the project manager shall make the final decision.

1.4 References

1. XGene 360 Software Requirement Specifications Document

1.5 Definitions, Acronyms, and Abbreviations

CVS	A version control system used for record the history of source files, documents, and allows developers to work on the same file concurrently.
IM	Instant messaging – the act of instant communication between two or more people over a network.
PDD	Program Design Document
PHP	Programming language primarily used on the Internet to provide server-side processing.
PP	Project Plan.
RED	Requirement Elicitation Document.
SDD	System Design Document.
XGene 360	The name of the application.

2. Management Structure

2.1 Project Lifecycle

For this project, we will be using the waterfall model; therefore, the major tasks can be broken down into the following phases:

Requirement Analysis
This phase defines the scope, problem, and constraints. This phase will involve active communication with the client in order to gather detailed problem definition. The deliverable for this phase is the software requirement specifications document.

Estimated timeframe: January 16 to February 6.

System Design, Program Design
This phase defines the system architecture, high level and low level design using software development techniques, patterns, and technologies. The deliverable for this phase is the design document which details how the application is implemented.

Estimated timeframe: January 30 to February 20.

Coding
This phase defines the implementation of the program in the chosen language as described in the design document. The deliverable for this phase is the code base of the application.

Estimated timeframe: February 20 to March 20.

Unit & Integration Testing, System testing, Acceptance Testing
This phase determines the correctness of the application. The testing will be performed first on individual components, then a few components as a group, and lastly as complete tests of the system. As the system becomes stable, user and client acceptance testing will be performed. The deliverable for this phase is the application providing the required functionality.

Estimated timeframe: Unit & Integration Testing: March 6 to March 31
System Testing: March 6 to March 31
Acceptance Testing: March 27 to March 31

The schedule listed above is subject to change.

2.2 Project Organization

2.2.1 External Interfaces

The external interfaces for this project will be the following:

Dr. Andrew Riseman – the client
Dr. Riseman is the sole client contact for input during the lifecycle of this project. The communication is done through the communication manager via e-mail unless a group meeting is arranged.

Roman Rudenko – the TA
He is a TA and supervisor for this team. The team will be reporting to the TA weekly to provide project status, tasks completion, further tasks, and any problems. Contact outside of the scheduled meeting will be done through the communication manager via e-mail.

2.2.2 Internal Structure

2.2.2.1 Roles and Responsibilities

Role	**Responsibility**
Project Manager	Responsible for overseeing the whole project, making executive decisions when needed, and resolving disputes, ensures that workload is distributed fairly.
Communication Manager	Responsible for maintaining the team members' contact information and for managing the asynchronous communication of the team, as well as for managing team-client communication.
Minutes Manager	Responsible for ensuring that meeting minutes are recording during meetings and posted to the project web site.
Configuration Manager	Responsible for performing forward traceability and reverse traceability.
Progress Manager	Responsible for keeping track of the progress of the project, for keeping the team informed of this progress, and for warning the team when tasks are lagging behind. Also, responsible for gathering the team members' worked hours each week.
Research and Training Manager	Responsible for managing the investigation and research of aspects related to the project and ensuring the team members are informed of the results of the research.
Risks Manager	Responsible for identifying and bringing to the team's attention potential risks as well as suggesting action or contingency plans to the team.

Software Versioning Control Manager	Responsible for providing the team with a means to control the various versions of the software and documents.
Web Master	Responsible for designing and managing the team’s web site.

2.2.2.2 Staffing

Role	Staff Member	Start Date	End Date
Project Manager	Jack Liu	Jan 4, 2006	Apr 8, 2006
Communication Manager	Susannah Poon	Jan 4, 2006	Apr 8, 2006
Minutes Manager	Gabriel Lee	Jan 4, 2006	Apr 8, 2006
Configuration Manager	Charles Bihis	Jan 4, 2006	Apr 8, 2006
Progress Manager	Jason Hui	Jan 4, 2006	Apr 8, 2006
Research and Training Manager	Gabriel Lee	Jan 4, 2006	Apr 8, 2006
Risks Manager	Charles Bihis	Jan 4, 2006	Apr 8, 2006
Software Versioning Control Manager	Jack Liu	Jan 4, 2006	Apr 8, 2006
Web Master	Jimmy Wong	Jan 4, 2006	Apr 8, 2006

2.3 Communication

Communication is a crucial key to the success of the project. E-mail and IM will be used on a daily basis to keep everyone’s understanding of the project and its progress up-to-date. Team groupware will be used in order to distribute and share information among members of the team. Verbal communication will be utilized on a weekly basis between group members to provide reports and clarify any issues that may arise.

The communication manager will also disseminate important information regarding newly provided information from the client and the TA.

Meeting schedule:
The current meeting schedule is twice as week: every Tuesday and Sunday from 2:00pm to 3:00pm.

2.4 Risk Management

Risks in the project will be managed by maintaining constant communication among members of the team. Weekly meetings, reports and e-mails will allow the team to identify risks early enough to be compensated for. In general, each risk will be given priority in discussion during a meeting. The risk manager will keep track of all risks that appear and record how each risk will be resolved. Risks will be scaled according to how much impact it will have on the project as a whole and the likelihood of it occurring. High impact, high occurrence risks will be given top priority, while low impact, low occurrence risks will be evaluated as to whether they will be dealt in the near future or not.

The risks identified below are the top five high impact risks with a high likelihood of occurring. Each risk will be given mitigation strategies for avoidance.

Risk	Semantics of any part of the design process is faulty or incompatible.
Type	Technical
Mitigation Strategies	• Have frequent meetings in order to reach a consensus and understanding of the overall design. • Clearly define conventions. • Maintain version control. • Maintain frequent developer communication.

Risk	Undesired behaviour of sister application, YellowLeaf, when running with product.
Type	Technical
Mitigation Strategies	• Run benchmarks for identification of inconsistencies. • Isolate any inconsistent functions and develop solution. • Study the design of the sister application for preventive knowledge.

Risk	Client is unsatisfied with application interface.
Type	Technical
Mitigation Strategies	• Frequently perform usability studies to rectify the design. • Develop alternative interfaces.

Risk	Realization of design takes more time and effort than scheduled.
Type	Managerial
Mitigation Strategies	• Schedule development to prioritize the implementation of basic functionality. • Maintain a simple design. • Cut out any features that will not be necessary to implement.

Risk	A developer becomes unavailable or cannot deal with the workload.
Type	Managerial
Mitigation Strategies	• Reassign workload so the developer will have a less critical workload. • If the workload is heavy, attempt to distribute it across other developers. • Maintain constant communication in order to detect risk earlier.

3. Planning and Control

3.1 Resource Identification

3.1.1 Staff

There are 6 (six) people working continuously on all phases for this project. No new staff members are expected to be added to the team at any time. Every resource will be involved in the project throughout the entire project.

3.1.2 Time

The application must be fully implemented and installed within 12 weeks. No extension will be possible due to the nature of this project.

3.1.3 Materials

The following software and hardware materials will be used for this project.

3.1.3.1 Hardware

Processor:	MAC G4 (dual processor)
Processor Speed:	1GHz
RAM:	1GB

3.1.3.2 Software

Operating System:	MAC OS X 10.2.8
Web Server:	Apache 1.3.33
PHP:	PHP 4.2.3
Database:	MySQL 3.23

3.2 Resource Allocation

3.2.1 Milestones

Jan 16 – Jan 23	Initial group meeting, review existing documents
Jan 23 – Jan 30	Begin Requirement Elicitation Document
Jan 30 – Feb 06	Complete first RED draft; Complete Project Plan
Feb 06 - Feb 13	RED and PP Due; begin SDD and PDD
Feb 13 – Feb 20	Complete first SDD draft; begin coding
Feb 20 – Feb 27	Continue on coding; begin testing and debugging
Feb 27 – Mar 06	Continue on coding; begin Unit Testing
Mar 06 – Mar 13	Complete coding; continue testing and debugging; begin Integration Testing and Overall System Testing
Mar 13 – Mar 20	Complete Testing & Debugging; Continue on Integration Testing and Overall System Testing

Mar 20 – Mar 27	Complete Integration Testing and Overall System Testing; Final Review of the application by the client
Mar 27 – Mar 31	Complete final coding refinements; Begin product deployment

3.2.2 Work Breakdown Structure

Tasks:

Documentation: Requirement Elicitation Document
Documentation: System Design Document
Design: Database Structure - Problem
Design: Database Structure – Plants
Design: Database Structure - Accounts
Design: UI
Design: User Privileges
Code: Error Logging
Code: Admin Interface
Code: Professor Interface
Code: TA Interface
Code: Student Interface
Code: Problem
Code: Algorithm – Plant Simulation
Code: Algorithm – Phenotype Expression
Code: Report Generation
Code: Evaluation
Code: User Creating/Modification/Deletion
Code: Problem Creating/Modification/Deletion
Code: Problem Assignment
Code: Password Change
Integration: UI with Code
Test: Functionalities
Test: Security
Test: System Testing
Documentation: Help Documentation
Documentation: Installation Documentation
Code: Installation Script
Test: Installation Script
Deliver: The Application

3.3 Tracking and Control

3.3.1 Cost

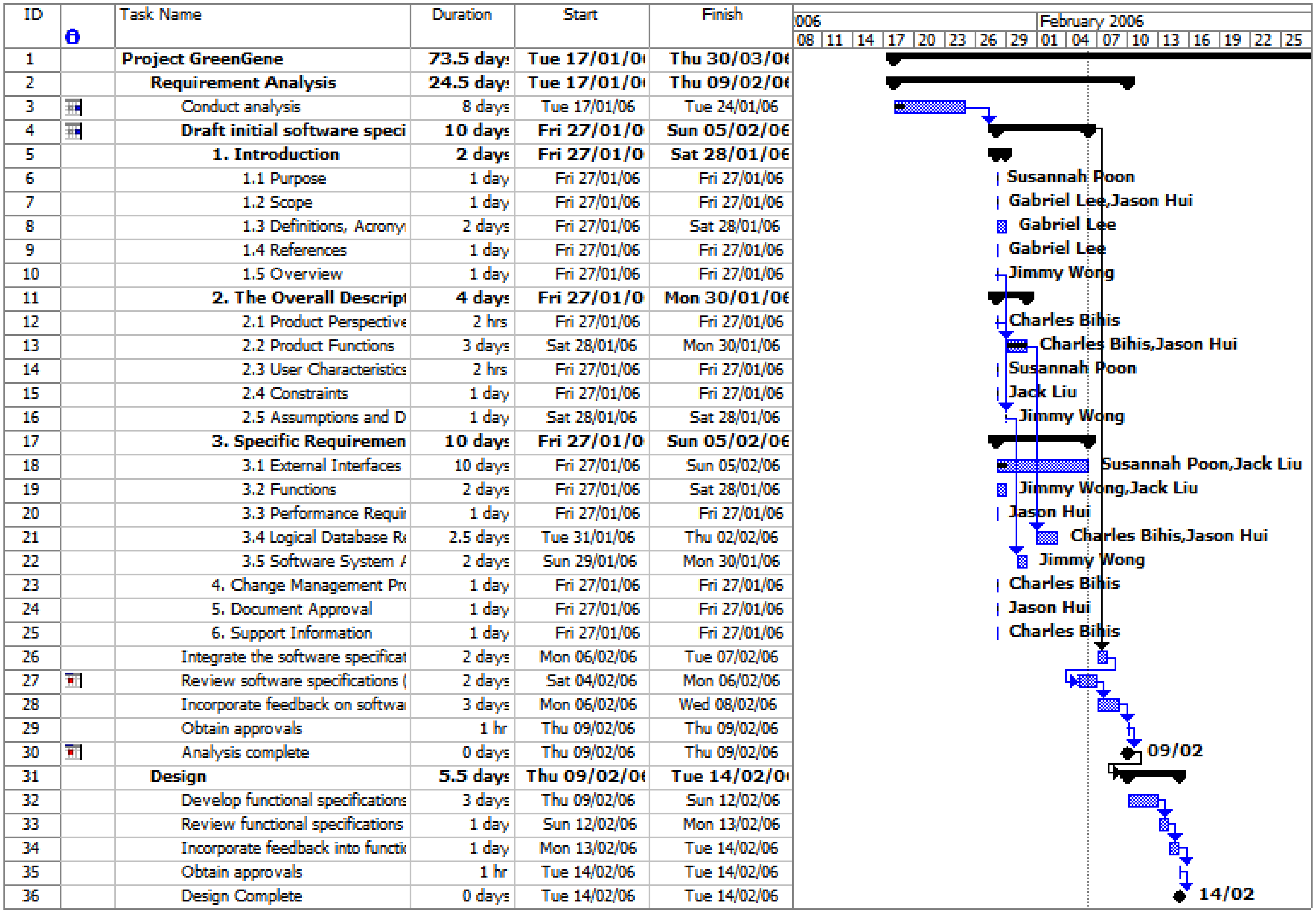
This project is under the GPL (Generic Public License) and is considered open source software. As a result, there is no allocated budget for this project.

No development tool requiring licensing or distribution fees will be used for the development of this project as the final release is to be open source and is covered by the GPL.

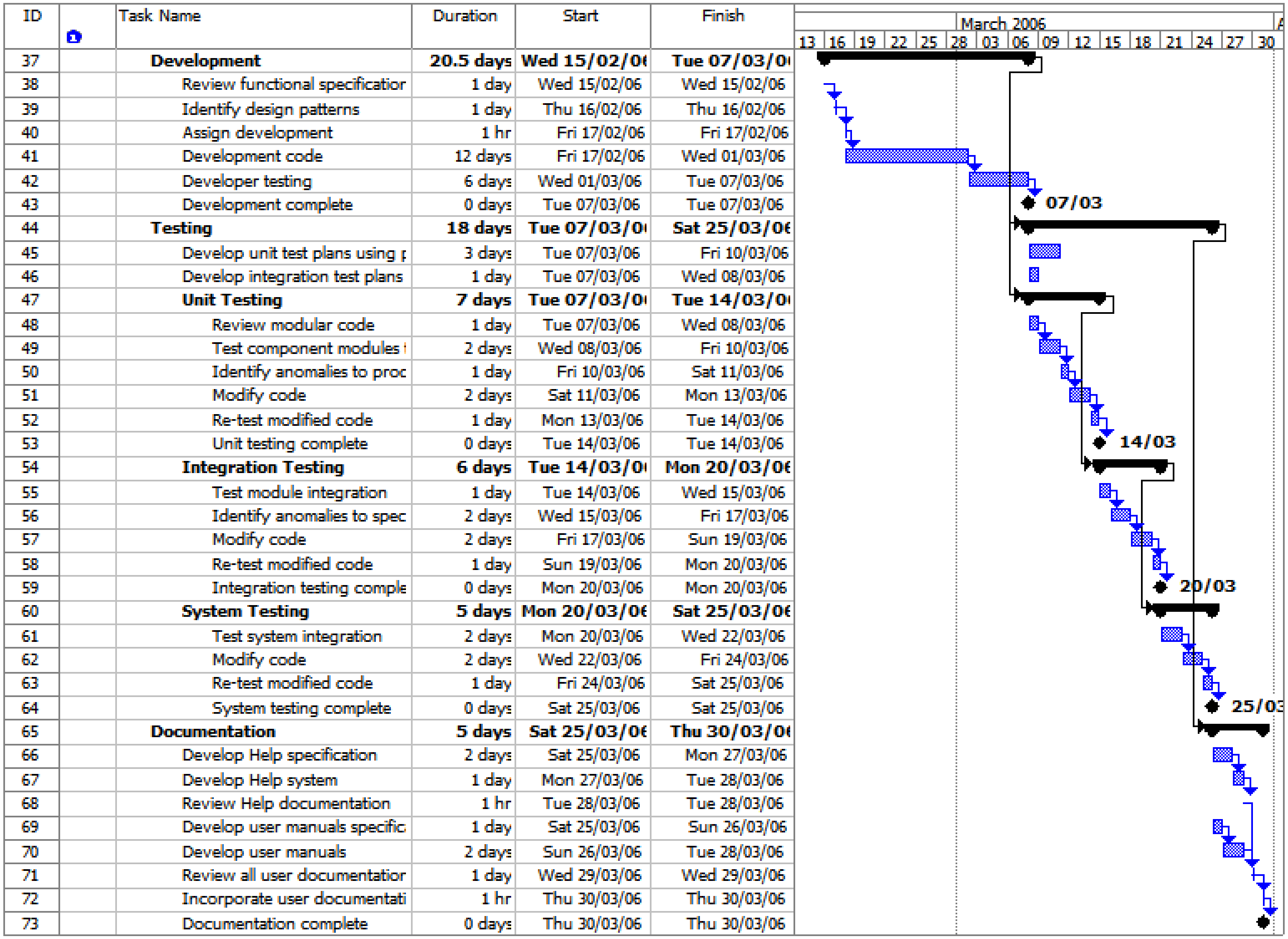
3.3.2 Schedule

The initial project schedule will be recorded with a Gantt chart using Microsoft Project.

The Gantt chart will be used to list the start and finish date of each scheduled task, the owner, and the estimated time to complete the task. Each day is assumed to be 2 (two) hours of working time.



ID	ⓘ	Task Name	Duration	Start	Finish
1		**Project GreenGene**	**73.5 days**	**Tue 17/01/0**	**Thu 30/03/06**
2		**Requirement Analysis**	**24.5 days**	**Tue 17/01/0**	**Thu 09/02/06**
3	▦	Conduct analysis	8 days	Tue 17/01/06	Tue 24/01/06
4	▦	**Draft initial software speci**	**10 days**	**Fri 27/01/0**	**Sun 05/02/06**
5		**1. Introduction**	**2 days**	**Fri 27/01/0**	**Sat 28/01/06**
6		1.1 Purpose	1 day	Fri 27/01/06	Fri 27/01/06
7		1.2 Scope	1 day	Fri 27/01/06	Fri 27/01/06
8		1.3 Definitions, Acrony	2 days	Fri 27/01/06	Sat 28/01/06
9		1.4 References	1 day	Fri 27/01/06	Fri 27/01/06
10		1.5 Overview	1 day	Fri 27/01/06	Fri 27/01/06
11		**2. The Overall Descript**	**4 days**	**Fri 27/01/0**	**Mon 30/01/06**
12		2.1 Product Perspective	2 hrs	Fri 27/01/06	Fri 27/01/06
13		2.2 Product Functions	3 days	Sat 28/01/06	Mon 30/01/06
14		2.3 User Characteristics	2 hrs	Fri 27/01/06	Fri 27/01/06
15		2.4 Constraints	1 day	Fri 27/01/06	Fri 27/01/06
16		2.5 Assumptions and D	1 day	Sat 28/01/06	Sat 28/01/06
17		**3. Specific Requiremen**	**10 days**	**Fri 27/01/0**	**Sun 05/02/06**
18		3.1 External Interfaces	10 days	Fri 27/01/06	Sun 05/02/06
19		3.2 Functions	2 days	Fri 27/01/06	Sat 28/01/06
20		3.3 Performance Requi	1 day	Fri 27/01/06	Fri 27/01/06
21		3.4 Logical Database R	2.5 days	Tue 31/01/06	Thu 02/02/06
22		3.5 Software System A	2 days	Sun 29/01/06	Mon 30/01/06
23		4. Change Management Pr	1 day	Fri 27/01/06	Fri 27/01/06
24		5. Document Approval	1 day	Fri 27/01/06	Fri 27/01/06
25		6. Support Information	1 day	Fri 27/01/06	Fri 27/01/06
26		Integrate the software specificat	2 days	Mon 06/02/06	Tue 07/02/06
27	▦	Review software specifications (	2 days	Sat 04/02/06	Mon 06/02/06
28		Incorporate feedback on softwa	3 days	Mon 06/02/06	Wed 08/02/06
29		Obtain approvals	1 hr	Thu 09/02/06	Thu 09/02/06
30	▦	Analysis complete	0 days	Thu 09/02/06	Thu 09/02/06
31		**Design**	**5.5 days**	**Thu 09/02/06**	**Tue 14/02/0**
32		Develop functional specifications	3 days	Thu 09/02/06	Sun 12/02/06
33		Review functional specifications	1 day	Sun 12/02/06	Mon 13/02/06
34		Incorporate feedback into functi	1 day	Mon 13/02/06	Tue 14/02/06
35		Obtain approvals	1 hr	Tue 14/02/06	Tue 14/02/06
36		Design Complete	0 days	Tue 14/02/06	Tue 14/02/06



ID	ℹ	Task Name	Duration	Start	Finish
37		**Development**	**20.5 days**	**Wed 15/02/0**	**Tue 07/03/0**
38		Review functional specificatior	1 day	Wed 15/02/06	Wed 15/02/06
39		Identify design patterns	1 day	Thu 16/02/06	Thu 16/02/06
40		Assign development	1 hr	Fri 17/02/06	Fri 17/02/06
41		Development code	12 days	Fri 17/02/06	Wed 01/03/06
42		Developer testing	6 days	Wed 01/03/06	Tue 07/03/06
43		Development complete	0 days	Tue 07/03/06	Tue 07/03/06
44		**Testing**	**18 days**	**Tue 07/03/0**	**Sat 25/03/0**
45		Develop unit test plans using p	3 days	Tue 07/03/06	Fri 10/03/06
46		Develop integration test plans	1 day	Tue 07/03/06	Wed 08/03/06
47		**Unit Testing**	**7 days**	**Tue 07/03/0**	**Tue 14/03/0**
48		Review modular code	1 day	Tue 07/03/06	Wed 08/03/06
49		Test component modules f	2 days	Wed 08/03/06	Fri 10/03/06
50		Identify anomalies to proc	1 day	Fri 10/03/06	Sat 11/03/06
51		Modify code	2 days	Sat 11/03/06	Mon 13/03/06
52		Re-test modified code	1 day	Mon 13/03/06	Tue 14/03/06
53		Unit testing complete	0 days	Tue 14/03/06	Tue 14/03/06
54		**Integration Testing**	**6 days**	**Tue 14/03/0**	**Mon 20/03/0**
55		Test module integration	1 day	Tue 14/03/06	Wed 15/03/06
56		Identify anomalies to spec	2 days	Wed 15/03/06	Fri 17/03/06
57		Modify code	2 days	Fri 17/03/06	Sun 19/03/06
58		Re-test modified code	1 day	Sun 19/03/06	Mon 20/03/06
59		Integration testing comple	0 days	Mon 20/03/06	Mon 20/03/06
60		**System Testing**	**5 days**	**Mon 20/03/0**	**Sat 25/03/0**
61		Test system integration	2 days	Mon 20/03/06	Wed 22/03/06
62		Modify code	2 days	Wed 22/03/06	Fri 24/03/06
63		Re-test modified code	1 day	Fri 24/03/06	Sat 25/03/06
64		System testing complete	0 days	Sat 25/03/06	Sat 25/03/06
65		**Documentation**	**5 days**	**Sat 25/03/0**	**Thu 30/03/0**
66		Develop Help specification	2 days	Sat 25/03/06	Mon 27/03/06
67		Develop Help system	1 day	Mon 27/03/06	Tue 28/03/06
68		Review Help documentation	1 hr	Tue 28/03/06	Tue 28/03/06
69		Develop user manuals specific	1 day	Sat 25/03/06	Sun 26/03/06
70		Develop user manuals	2 days	Sun 26/03/06	Tue 28/03/06
71		Review all user documentatior	1 day	Wed 29/03/06	Wed 29/03/06
72		Incorporate user documentati	1 hr	Thu 30/03/06	Thu 30/03/06
73		Documentation complete	0 days	Thu 30/03/06	Thu 30/03/06

3.3.3 Quality and Functionality

After each phase, the configuration manager is responsible to ensure the application meets the requirement standard outlined in the Software Requirement Specifications document and is implemented to the specification according to the System Design Document.

The quality of code will be maintained by using a software versioning control system as described in Section 4.1 of this document and quality of the application is maintained through the testing as described in 4.2.

Team members will record their individual accomplishments for the week and send the report to the project manager. The project manager will include the reports in the aggregated weekly status report.

Each member will be responsible for tracking the amount of time they spend on various tasks. The progress manager will be responsible for tracking the total amount of time spent on each task. The project manager will be responsible for updating the schedule based on the actual progress.

4. Supporting Plans

4.1 Configuration Management

The team will be using Perforce (Concurrent Versioning System) to manage the project data. Each developer will use a Perforce client to access the file repository.

All files, code, and documents required for the application implementation will be stored on the Perforce server where file versioning and history will be maintained. For a text file, multiple checkouts are allowed; if conflict happens during the submission, the user will be responsible for merging the files to make sure the functionality checked in by the other developer is not lost.

All code that is will be checked into the Perforce server will be tested by the developers to make sure it does not break the build. A build of all files that is checked into CVS will be performed at least once a week to ensure that the current build is working.

4.2 Quality Assurance

In order to achieve desired functionality, the implementation will be done exactly according to the detailed design documents. Every developer will have their code reviewed by a peer prior to checking in to ensure the implementation is correct and is based on the design document.

All bugs will be logged into a bug database. Each bug will be assigned to the corresponding phase leader, who will then assign it to the appropriate developer for repair. Bugs will be prioritized and will be marked as resolved when it is fixed.