Software Project Management Plan

for *Earth* Project

Alex Egan
Callum Baillie
George Sainsbury
Hill-Jian Huang
Ming Jie Tan
Jonathan Velasco
Sahil Choujar
Yang Qing
Ken S'ng Wong
Filimoni Lutunaika
Mohammad Bamogaddam
Kun Zhou
Xiaodong Cui
Hemant Singh

May 11, 2008

Contents

1 Introduction

1.1 Purpose and Scope

This project endeavours to build upon the existing framework that has already been developed for *Earth*. Currently there exist a number of features that would be desirable to be implemented in this open source software. It is our aim to implement as many features as is necessary to further evolve Earth into a highly manageable and efficient tool.

This document outlines the direction the project will take to best deliver the final product. In other words, this document will show how our team is assigned to deal with the task required of us. This project management plan looks at how we will deal with risks and how they will be mitigated and how they will be controlled should they arise. Furthermore, we show how we intend to develop additional features of Earth, showing not only the process model that we will use but also how the work is scheduled to take place.

1.2 Project Deliverables

The following section outlines when specific documents and deliverables will be completed and available. These are the main deliverables. However, more detailed information about internal milestones are provided in Section 7.

1.2.1 First Document Drafts

1. Software Project Management Plan - due 31/03/08

1.3 Evolution of the Plan

This plan will be reviewed on a weekly basis for the duration of the project. In addition to these regular updates, every time our work on the project differs to the proposed workplan, this document will be updated as necessary. These changes may be made by any member of the group. To ensure that the project stays on track, along with this document, there are numerous risk management procedures in place. Furthermore, with regular meetings, we are able to stay aware of where we are compared to where we should be as set out by our milestones and deliverables.

2 References

Software Engineering 8th Ed., I. Sommerville, Addison-Wesley, 2007

3 Definitions

Following is a table of acronyms and their meanings.

CASE	Computer Assisted Software Engineering
SVN	Subversion
SPMP	Software Project Management Plan

Table 1: Acronyms

4 Project Organisation

4.1 Roles and Responsibilities

This project group has been divided up such that we have three development teams. Each team will be assigned their own tasks to complete, and testing of each individual component will be completed by the group that it was assigned to. This group structure allows for the members of each individual group to focus on development of their tasks and not have to worry about dealing with organisation.

Minutes: Ken S'ng Wong

George has taken on the task of taking and managing the minutes for all formal meetings.

Git Management Team: Alex Egan, Ken S'ng Wong, Mohammad Bamogaddam

This group manages the git repository structure for the entire group as well as maintaining each development team's repository and the document repository.

Primary Meeting Chairperson: Ming Tan

Ming is organising agendas and chairing the meetings. However at times other members may take on his role, in the event this does happen, it will be noted in the agenda.

Development Team 1 - Gui and Testing Development

Development Team 1 consists of George Sainsbury, Alex Egan, Xiaodong Cui and Filimoni Lutunaika.

Development Team 2 - Plugin Development

Development Team 2 consists of Ming Tan, Callum Ballie, Kun Zhou, Mohammad Bamogaddam and Yang Qing. Development Group 2 will also facilitate the documentation of this project.

Development Team 3 - Database Development

Development Team 3 consists of Jonathan Velasco, Sahil Choujar, Ken S'ng Wong, Hemant Singh and Hill-Jian Huang.

5 Risk Management Plan

Detailed below are various risks that may affect the project and strategies for dealing with them.

5.1 Staff unable to work due to sickness

- **Likelihood:** Moderate as Staff are vulnerable to various diseases such as the flu.
- **Severity:** It could be catastrophic if the absence is for a long period of time as some staff possess specialised knowledge in specific areas. However, if the time of absence is short, the severity will be tolerable.
- **Indicators:**
  - Advanced notice from the staff member who will be unavailable or unable to work
  - Staff member does not attend meetings.
  - Staff member complains about their health
- **Strategies:**
  - Organise staff to work in teams so that more than one person understands and can work on each task. This allows the team to continue working even if a member is unable to work.
  - Provide time slack on all tasks so that other members have enough time to pick up the jobs of those unable to work.

5.2 Staff unable to work due to other commitments

- **Likelihood:** Moderate as staff may have unexpected commitments such as business or family trips.
- **Severity:** It could be catastrophic if the absence is for a long period of time as some staff possess specialised knowledge in specific areas. However, if the time of absence is short, the severity will be tolerable.
- **Indicators:**
  - Advanced notice from the staff member who will be unavailable or unable to work
  - Staff member does not attend meetings.
- **Strategies:**
  - Organise staff to work in teams so that more than one person understands and can work on each task. This allows the team to continue working even if a member is unable to work.
  - Provide time slack on all tasks so that other members have enough time to pick up the jobs of those unable to work.

5.3 Difficulty in adding features due to poor software design in existing code

- **Likelihood:** Moderate as the code was likely developed to meet existing needs without foresight for unexpected but desired improvements.
- **Severity:** High as the time necessary to fix any design issues in the existing code to implement certain features could increase significantly.

- **Indicators:**
  - Problems when attempting to design a feature to be implemented around the existing code.
- **Strategies:**
  - Attempt to avoid features that may not be easily integrated with the existing code.
  - Redesign of certain sections of existing code to implement new feature.

5.4 Unable to contact Rising Sun Pictures to Clarify Requirements

- **Likelihood:** Medium as the main developers of earth are situated at the Sydney office.
- **Severity:** Medium because it will not allow for certain requirements that require clarification to proceed until contact is made.
- **Indicators:**
  - Lack of email response.
  - Advanced notice that their offices are busy.
- **Strategies:**
  - Proceed to other already clarified requirements.
  - Prepare numerous requirements to clarify during meetings with Rising Sun Pictures.

5.5 Lack of knowledge of Ruby, Rails, Git

- **Likelihood:** The likelihood is high as all of the above tools are new to the development team.
- **Severity:** The severity is catastrophic as it would be extremely difficult to finish the project on time and with a high level of quality without knowledge of the tools.
- **Indicators:**
  - Slow adoption of required set of tools upon selection
  - Staff member complains about any of these tools
  - Selection of these tools late, having reduced time to study them
- **Strategies:**
  - Inform the development team that these are the tools to be used in advance so that they have time learn to use them.
  - Organise a tutorial or lectures of these tools and its functionality to show the developers how to use the tools and what they can be used for.
  - Include time in the schedule for learning how to use these tools and their functionality.

5.6 Requirements not completed on time - excessive number of requirements

- **Likelihood:** The likehood is moderate as the requirements selection is done internally.
- **Severity:** High as features intended for the final product will not be completed.
- **Indicators:**
  - The number of requirements is too large.
  - Clients continue adding more requirements
  - Lack of sufficient communication between the development team and the clients causing the wrong requirements to be implemented.
- **Strategies:**
  - Have the development team and the clients meet regularly to review the requirements and the progress of their implementation.
  - Maximize the detail of the requirements in the SRS.
  - Develop the project in such a way that adapting to new or changed requirements is easy.

5.7 Requirements not completed on time - unreachable complexity of the requirements

- **Likelihood:** The likehood is high as it is likely that after delving into the existing code, that implementing a feature could become a much larger problem.
- **Severity:** High as the system may be unable to work correctly or may behave in an incorrect or inefficient manner.
- **Indicators:**
  - The level of difficulty of the implementation of the requirements is too high.
  - Lack of sufficient communication between the development team and the clients causing the wrong requirements to be implemented.
- **Strategies:**
  - Have the development team and the clients meet regularly to review the requirements and the progress of their implementation.
  - Maximize the detail of the requirements in the SRS.
  - Develop the project in such a way that adapting to new or changed requirements is easy.

5.8 Underestimate the development time to reach each milestone

- **Likelihood:** The likelihood is medium as initial estimates of task time may be incorrect, and result in needing more time.
- **Severity:** High as the project may be not delivered on time or may not reach the required quality level.
- **Indicators:**
  - Milestones not met on time.

  - Requirements changing constantly.

- **Strategies:**
  - Increase the free slack time of each milestone in order to have some extra time if needed.
  - Reassign members/development teams to increase the speed at which the core components are completed.

5.9 New features of the project do not integrate correctly with Earth

- **Likelihood:** The likelihood will depend on the level of integration required and how interconnected a component is. Therefore, the likelihood can vary from low to high.
- **Severity:** Moderate as time to analyse and fix problems may delay the feature being implemented
- **Indicators:** Feature doesn't work as intended with Earth
- **Strategies:**
  - Have input from multiple developers when designing components
  - Begin integration early so that there is more time to deal with any problems that occur
  - Have incremental integration with smaller components so that problems are easier to identify

5.10 Unable to work at the computer labs

- **Likelihood:** Medium because while there are many labs, not all systems will have the necessary tools for this project.
- **Severity:** Low because it is not required to work in the computer labs.
- **Indicators:**
  - Advance notice that the computer labs will not be available.
  - No computers are available due to maintenance or use by other people
- **Strategies:**
  - Install the required tools on other computers or use remote access to computers that have the required tools.
  - Install the required tools on computers in the SEP Labs.

5.11 Central Earth Git repository unavailable

- **Likelihood:** Low as maintenance of the Git repository and its availability is the responsibility of a separate department.
- **Severity:** Low as each developer has their own copy of the Git repository to work on.
- **Indicators:**
  - Degraded performance.
  - Scheduled downtime or maintenance.
- **Strategies:**
  - Continued Development on local copies of the repository, with progress to be pulled/pushed once the repository is restored.
  - Keep the number of files to a minimum by splitting work into groups and sections.

6 Process Model

Agile Process will be adopted for this project. This entails short development cycles, with many milestones to be met. In general we will adopt a 6 week development sprint, where requirements investigation and prototyping will occupy 1 week, with 3 weeks of development time and 1-2 weeks of testing time. This allows for developed and tested products and endorses incremental development.

7 Work Plan

This project makes use of an iterative process where the existing code will be improved upon with additional features.
The time estimates provided show the average duration of time that each member assigned to the task spends on that task.

7.1 Milestone 1 - Work Activities

- Infrastructure
- Documentation
- Ticket 27
- Ticket 66
- Ticket 75
- Ticket 120
- Ticket 127
- Ticket 145
- Testing

7.1.1 Infrastructure

This task involves the installation of Earth along with ruby and rails, and ensuring that any code changes can be integrated, compiled and tested. As such it is to be undertaken by all group members.

Resource: All Development Staff Time Estimate: 5hrs

7.1.2 Documentation

This involves the development and maintenance of existing documents for the project plan and any subsequent documents.

Resource: Ming Tan Time Estimate: 8hrs

7.1.3 Ticket 27

This ticket involves figuring out and recording of the filetypes of files in the database.

Resource: Development Group 3 (Jonathan Velasco, Sahil Choujar, Ken S'ng Wong, Hemant Singh and Hill-Jian Huang) Time Estimate: 4hrs

7.1.4 Ticket 66

This ticket involves the listing of space used by each user.

Resource: Development Group 3 (Jonathan Velasco, Sahil Choujar, Ken S'ng Wong, Hemant Singh and Hill-Jian Huang) Time Estimate: 4hrs

7.1.5 Ticket 75

This ticket involves the sorting of directory and file listings by name and size.

Resource: Development Group 1 (George Sainsbury, Alex Egan, Xiaodong Cui, Filimoni Lutunaika) Time Estimate: 4hrs

7.1.6 Ticket 120

This ticket involves the display of green bars for sizes in the "All Files" view.

Resource: Development Group 2 (Ming Tan, Callum Ballie, Kun Zhou, Mohammad Bamogaddam and Yang Qing) Time Estimate: 4hrs

7.1.7 Ticket 127

This ticket involves the display of a warning should the user's browser not be SVG capable when attempting to view SVG.

Resource: Development Group 2 (Ming Tan, Callum Ballie, Kun Zhou, Mohammad Bamogaddam and Yang Qing) Time Estimate: 4hrs

7.1.8 Ticket 145

This ticket requires the relabelling of file sizes from " $0TB"tosomethingmoremeaningful.$

Resource: Development Group 1 (George Sainsbury, Alex Egan, Xiaodong Cui, Filimoni Lutunaika) Time Estimate: 4hrs

7.1.9 Testing

Testing will be conducted on each of the tickets above, and testing scripts or unit tests will be created as necessary.

Resource: All Development Groups Time Estimate: 8hrs

7.1.10 Milestone 1 - Schedule Allocation

The milestones and tasks are shown graphically in Figure 2 below. This figure shows the relative times between the deadlines of the tasks required and also shows the estimated time for the completion of each individual tasks.

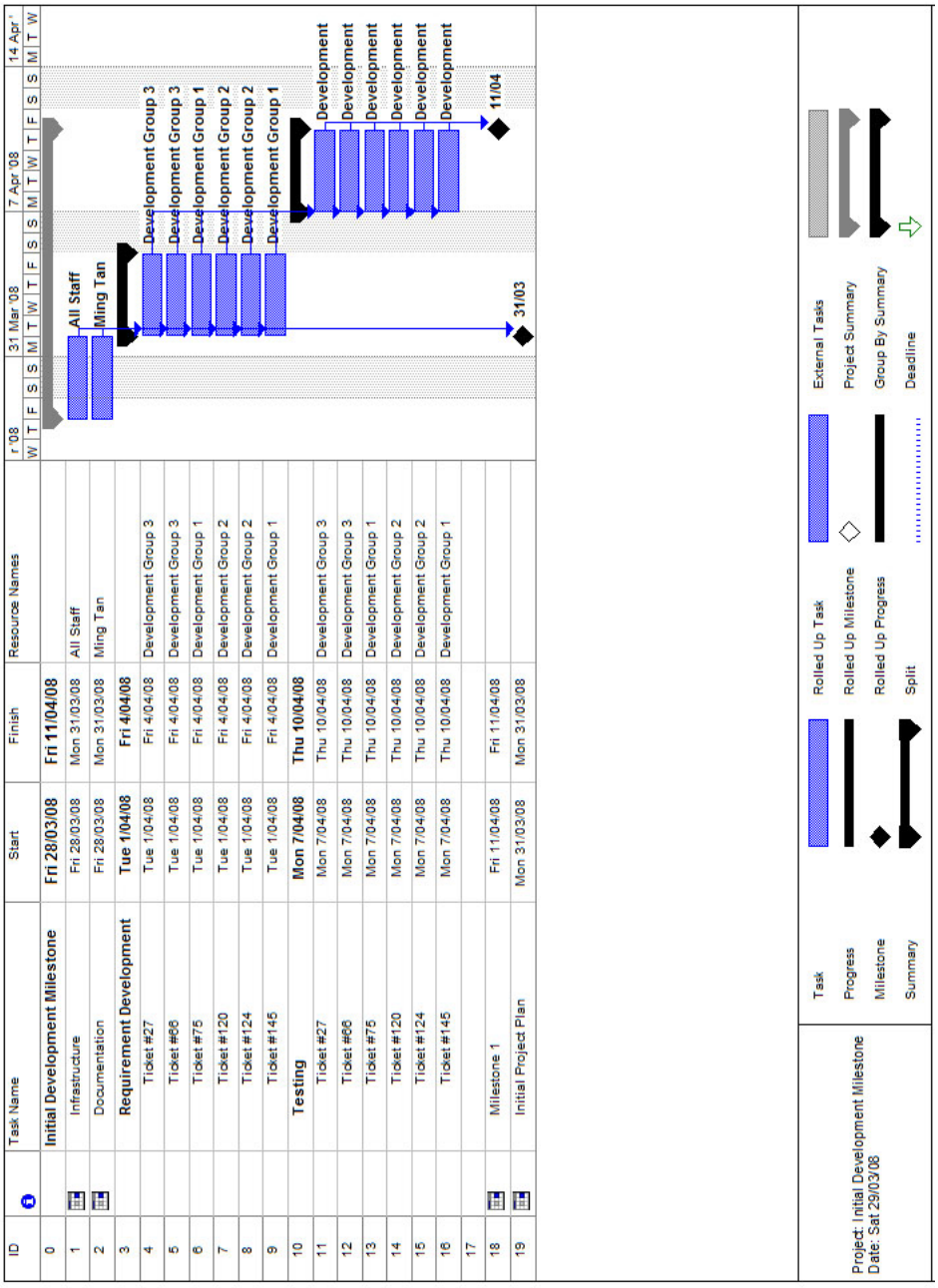

Figure 1: Gantt chart of project tasks for mileston 1

7.2 Milestone 2 - Work Activities

- Ticket Investigation
- Documentation
- Ticket 138
- Ticket 146
- Ticket 42
- Ticket 23
- Database Design
- Plugin System Design
- Testing

7.2.1 Ticket Investigation

Each subgroup took various tickets, and investigated what would be needed to complete each ticket. This was done in order to discover which tasks would be best to undertake for this development iteration. Tickets 23, 40, 42, 138, 146 and 169 were investigated.

Resource: All Development Staff Time Estimate: 5hrs

7.2.2 Documentation

This involves the development and maintenance of existing documents for the project plan and any subsequent documents. During this iteration, the project plan was updated as well as documentation for testing processes and the git repository structure being added.

Resource: Ming Tan, Alex Egan, Mohammad Bamogaddam Time Estimate: 8hrs

7.2.3 Ticket 138

This ticket involves the testing of the existing improvements made to the radial and treemap views with filters to determine whether improvements are necessary.

Resource: Development Group 1 (George Sainsbury, Alex Egan, Xiaodong Cui, Filimoni Lutunaika) Time Estimate: 4hrs

7.2.4 Ticket 146

This ticket involves the addition of a script to allow for continuous testing of the code with test cases.

Resource: Development Group 1 (George Sainsbury, Alex Egan, Xiaodong Cui, Filimoni Lutunaika) Time Estimate: 4hrs

7.2.5 Ticket 42

This ticket involves the addition of job name, sequence name and shot name as metadata to allow for file searches via this information.

Resource: Development Group 1 (George Sainsbury, Alex Egan, Xiaodong Cui, Filimoni Lutunaika) Time Estimate: 4hrs

7.2.6 Ticket 23

This ticket involves the tagging or folksonomy of files and directories so that earth will be able to understand in a generic way the "RSP" file system concepts.

Resource: Development Group 3 (Jonathan Velasco, Sahil Choujar, Ken S'ng Wong, Hemant Singh and Hill-Jian Huang) Time Estimate: 4hrs

7.2.7 Database Design

This task involves reverse engineering of the existing database structure, and documenting it to allow for system changes to be made more easily.

Resource: Development Group 3 (Jonathan Velasco, Sahil Choujar, Ken S'ng Wong, Hemant Singh and Hill-Jian Huang) Time Estimate: 4hrs

7.2.8 Plugin System Design

This task involes the reverse engineering of the existing use of plugins in the system, and documenting how it is currently done and how it could best be structured.

Resource: Development Group 2 (Ming Tan, Callum Ballie, Kun Zhou, Mohammad Bamogaddam and Yang Qing) Time Estimate: 4hrs

7.2.9 Testing

Testing will be conducted on each of the tickets above, and testing scripts or unit tests will be created as necessary.

Resource: All Development Groups Time Estimate: 8hrs

7.2.10 Milestone 2 - Schedule Allocation

The milestones and tasks are shown graphically in Figure 2 below. This figure shows the relative times between the deadlines of the tasks required and also shows the estimated time for the completion of each individual tasks.

7.3 Milestones

7.3.1 Initial Features Update

This milestone allows for the group members to familiarise themselves with the Earth code as well as the tools required to develop for Earth. In addition it allows a test run of the development sprint.
Completion criteria: The milestone has been achieved if tickets 27, 66, 75, 120, 127 and 145 are implemented and tested.
Due: 11/04/08

7.3.2 Development Sprint 2

This milestone allows for more in depth tasks to be undertaken. In addition, design documentation is
Completion criteria: The milestone has been achieved if tickets 23, 42, 138, and 146 are implemented and tested. In addition, there must be some design documentation for both the database

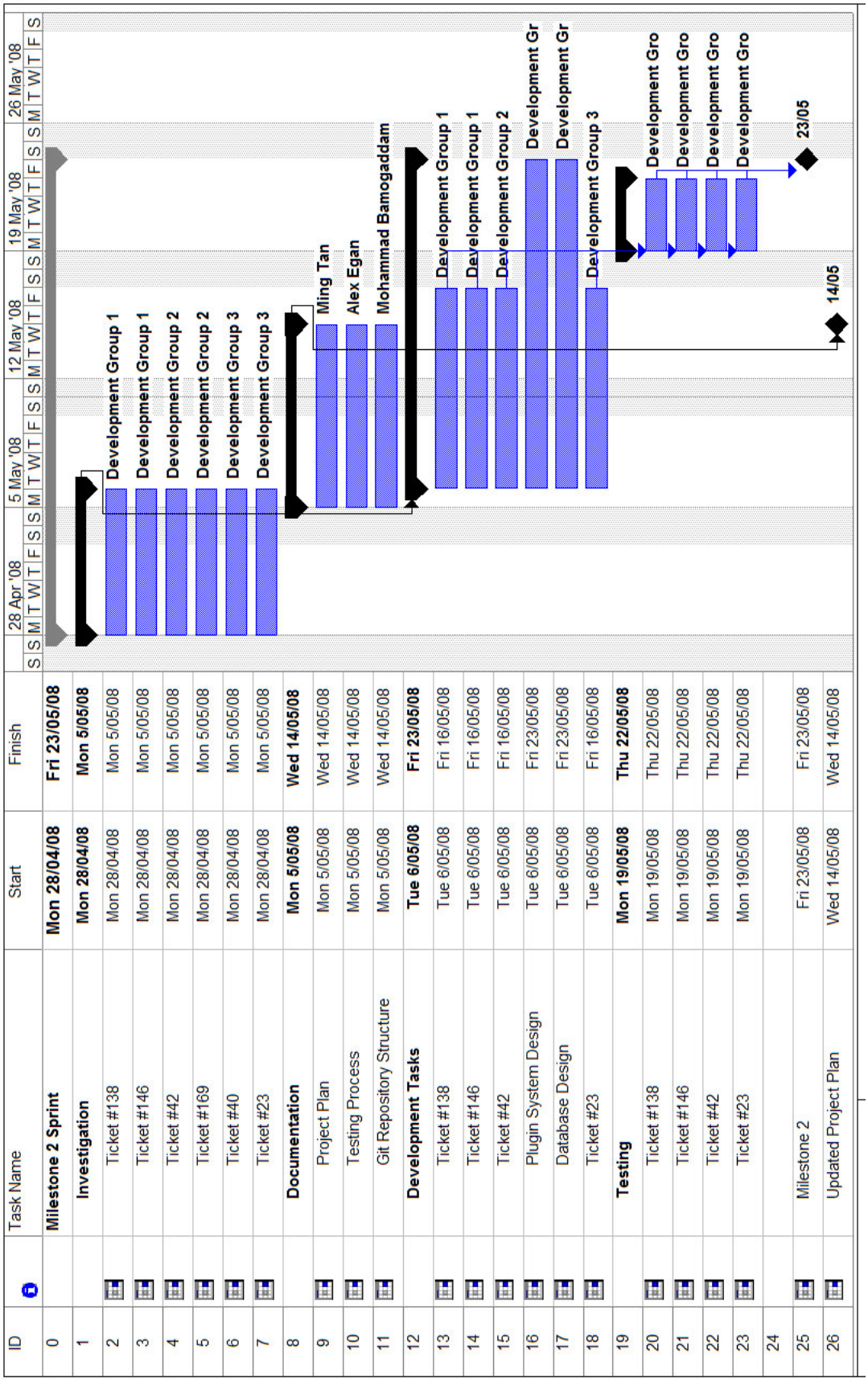

Figure 2: Gantt chart of project tasks for milestone 2

and plugin systems.
Due: 23/05/08

7.4 Resource Allocation

As described in Section 4.1, people will be allocated to each task in the following fashion:

- **Development Group 1:** George Sainsbury, Alex Egan, Xiaodong Cui, Filimoni Lutunaika
- **Development Group 2:** Ming Tan, Callum Ballie, Kun Zhou, Mohammad Bamogaddam and Yang Qing
- **Development Group 3:** Jonathan Velasco, Sahil Choujar, Ken S'ng Wong, Hemant Singh and Hill-Jian Huang
- **Documentation:** Ming Tan, Alex Egan, Mohammad Bamogaddam

If a task is completed early, or it is noted that less people are required to complete a task on schedule, they will be reallocated to other tasks.

8 Supporting Plans

8.1 Git Repository Process

8.1.1 Human Resources Allocation:

- Each group will have a GitHub leader who is responsible for their group's copy of the central repository.
- The GitHub leaders will be responsible for the central repository.

8.1.2 Creation:

- At first, the GitHub manager should fork the origin from RSP repository. So, we will have: manager/earth
- Then, GitHub leaders should fork (clone) their masters from the origin (i.e. manager/earth). So, we will have: leader1/earth, leader2/earth and leader3/earth
- each leader should give all his group members permitions to push their group fork
- Then, each group member should only clone the group's master fork to his local PC (i.e. leader1/earth, leader2/earth and leader3/earth).

8.1.3 Upon Completion of a task:

- He pulls the latest changes from the group's master. (git pull)
- Test his task locally.
- Pushes the changes.
- Sends a pull request to the group GitHub leader (and all other group memebers). This request should include (at least): related task number or description, files changed, related ticket numbers. (this really a notification not a request since he already pushes his changes)

8.1.4 Upon pull request to Github Leader:

- The group GitHub leader reviews the changes in terms of: documentation standards, etc.
- Performs the automated test to make sure that nothing is corrupted. (he could fully test the changes and approve it "for discussion in the meeting")
- If the changes are approved: Sends a pull request to: all group members GitHub manager about the changes: related task number or description, files changed, related ticket numbers and person responsible.
- If the changes are not approved: Reverts back the group's master to its status before these changes are committed, and sends an email to the group member clarifying the reasons of rejecting the changes.

8.1.5 When the GitHub manager receives a pull request from a GitHub leader:

- The Github manager pulls the latest changes from that group's master. (this could includes any necessary merges)
- The GitHub manager performs an automated final test. (should be specified)
- If it passed the test successfully, sends a pull request to all other GitHub leaders so that they pull the latest changes to their forks. (He could marks a version??)
- If it does not pass the test: Reverts back the origin to its state before these changes, and inform the group GitHub leader by an email about the reasons. (or from the beginning, make a fake fork or branch for testing)

8.1.6 Milestone Completion:

- The GitHub manager marks a version/release
- He sends a pull request to Earth people. This should contain: Milestone features, (for discussion)