Group A
Software Engineering C
Massey University
2008

RoadMate

8 – 8 – 2008

Lorem Ipsum

Summary

RoadMate is a web application designed to assist avid travelers in organising and coordinating car pools. As fuel prices continue to sore, our hope is that RoadMate will become an intricate part of New Zealander's travel planning process – Saving fuel, money and the environment. RoadMate is currently under development at Massey University as a 3rd year software engineering project.

Project Plan

Design Methodology

We were eager to take advantages of the benefits of the agile development techniques we have learned. Yet the design methodology we have adopted is one less agile and similar to RUP, this is for several reasons:

- Our development team is large (for the size of the project) and inexperienced. Agile development works well with small teams of experienced developers. Maintaining a repository of formal documentation, visual modeling techniques, a strong emphasis on project management all support general understanding of our model of the system, and help to keep us on track despite our inexperience.

- Skill and experience with the development platform varies significantly between team members. Agile approaches are dependent on all team members understanding the technology and being able to work relatively independently. We realize it is better for us to have one or two experts who fully understand the technology to develop the framework and teach the rest of us, than to have six equally inexperienced team members, all trying to find their way in the dark.

- There are constraints on reporting and demonstration that we as a team don't have control over. Although we can choose the development process to use, we can't avoid having to produce the project or weekly reports and the design documentation (Gantt charts, RMMM plans) that they should contain.

- We can be confident that the requirements are fully specified. We aren't dealing with a client who doesn't fully understand the system they want us to develop, so we don't need to rely on the agile techniques (like on-site client, feedback or revisions to the core requirements) that we would need if we were less certain about the requirements.

The method we have employed is a mix of aspects of agile and feature-driven development, and some Big Design Up Front practices, like RUP. It is our attempt at applying the industry best-practices that can meet the unique constraints of this project.

- **Iteration and prototyping** – seeking "client feedback" on delivered functionality

- **Prioritizing requirements** and only implementing the top few, per iteration.

- **Use-case driven development**, each piece of functionality relates to a use-case, which in turn relates to the requirements

- **Model visually** – maintaining an ERD model of the system, so we all understand the entities and their members.

- **Maintaining documentation** – using Google Docs to collaborate on public working documents like meeting agendas, and using Subversion to store a version-tracked public repository of all design docs.

- **Code is double-checked** by a partner (not as extreme as **pair programming**; but coding as a group of two or three and rarely making changes alone).

- **Regular reports and meetings** to track the status of tasks and issues, and to report to the team manager.

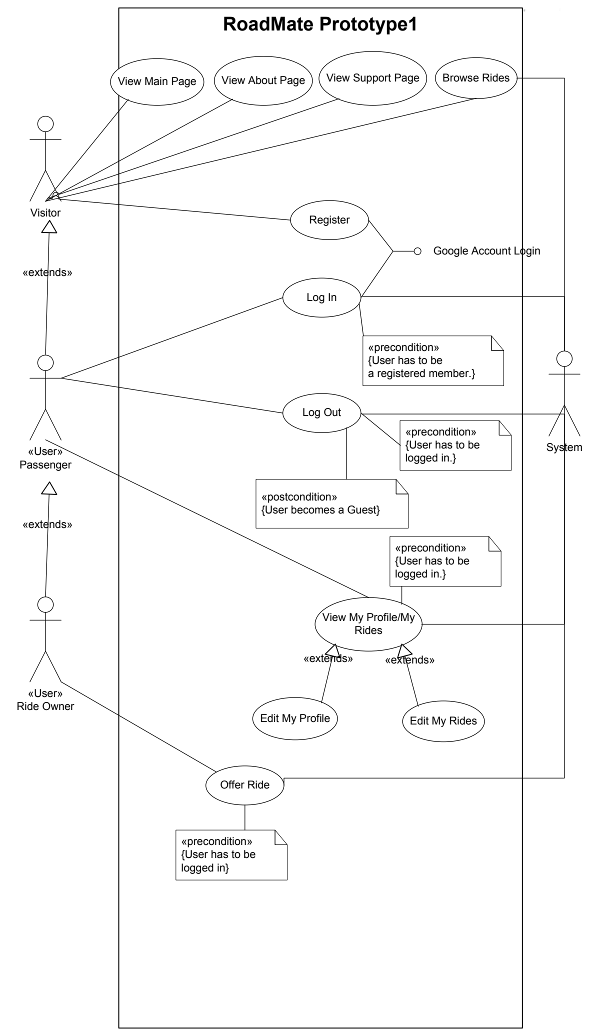
Risk Mitigation, Monitoring and Management Plan

Name	Probability	Category	Description	Impact	Monitoring	Proactive Actions	Reactive Actions
Project goes over time	Low	Business	A possibility is that the team doesn't manage to finish the project on time, can happen as a result of lack of preparation.	Medium	Develop a Gantt chart that has set milestones and decision points at critical stages which will help us identify our Progress.	Keep modifying and updating the Gantt chart so that it meets the deadline.	Renegotiate deadline with the client, explaining why that is the case.
Loss of personnel	Low	Project	One or more team members decide to leave the team.	High	Monitor team members to see if they seem likely to leave.	Get a couple of week early notice from the team member before he/she leaves.	Hire a back up team member whom have been interviewed already and ready to work.
Authentication system fails or has flaws.	Low	Technical	Design flaws in the system allows for unauthorized access to the system.	High	Regularly check for unauthorized access to the system.	Adopt the Google authentication system, since it is a more reliable system.	Assign a team member or two to develop our own version of the authentication system.
End product has usability issues	Low	Technical	User find it hard to understand the GUI, and find it hard to navigate the website.	Medium	Adopt good HCI practices.	Test the GUI before the final deployment. Test must include a wide range of user (novice and experts).	Understand the difficulties the user is facing and modify the UI in accordance to that.
Project requirements change	Medium	Project	Client changes the requirements sometime during development.	Medium	Regularly check progress against Requirements.	Try and maintain regular communication with the client.	Check if the new requirements require re planning and assign tasks to the team.
People don't work well together	Medium	Staff	Team members fail to communicate together	Medium	Team leader to find out early on how does the team look and decided whether a change is needed.	Get regular feedback from members to see if there are problems.	Find out what's wrong, try and solve, if can't hire and fire members as needed.
Lack of team Knowledge	High	Staff	The team might lack someone who specializes in a particular field.	Low	Team leader to know the team very well and know their relative experiences.	Before the implementation phase check whether the team has the knowledge to accomplish all the given tasks.	Seek knowledge; assign task to a member and the member needs to learn what's needed to accomplish the task.
Problems with technology	Medium	Technical	We will be using some new technologies and there is the possibility that we struggle and don't mange to learn the new tech.	Medium	Team leader to continuously check with team member and see how comfortable they are with the new techs, and if they aren't find out what's bothering them and help them to manage it.	Before choosing the technology make sure that team members are confident in using the technology and they have no problem learning it. Also have a plan B, i.e. another technology that can be used.	Find out the problems that are faced and try and solve them, otherwise change the technology to the plan b technology.

Design Documentation

Use Case Diagram

To the right is the use case diagram for our first prototype, with the actors listed below.



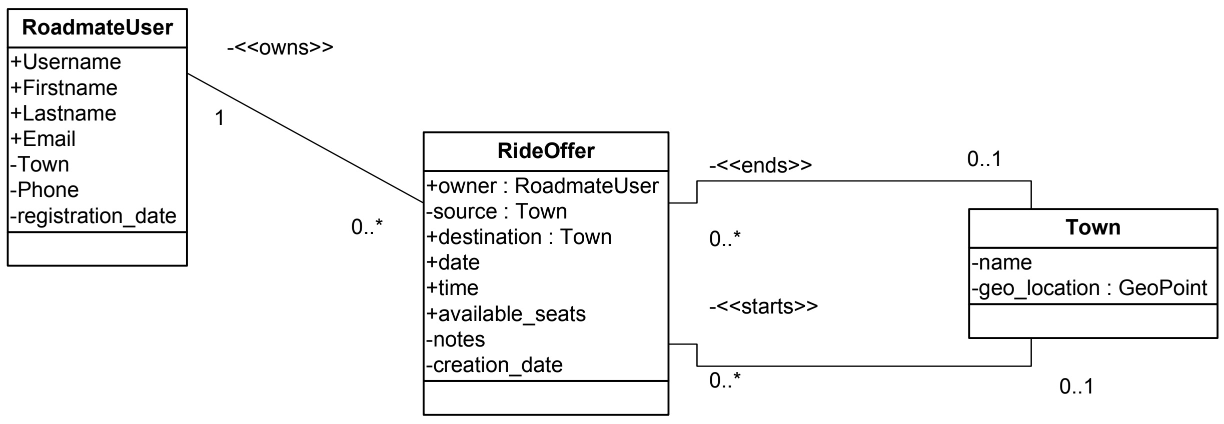
Actors

Visitor – Any visitor to the RoadMate website.
User – A registered user.
Ride Owner – Owner of an offered ride.
System – The RoadMate system.

Use Case Descriptions

The individual use case descriptions in standard template form have been provided in the appendix of this report.

Entity – Relationship Model



Project Scope

Terminology

The following is brief outline of the terminology used within the project scope section.

Visitor – A person viewing the site, who may or may not be logged in.
User – A person viewing the site, who **is** logged in.

Prototype One

The following is an outline of the site structure and functionally as provided in the current iteration of prototype one.

- **Home**
  The home page acts as the main gateway to the rest of RoadMate.
  - Welcomes visitors and provides a brief description of RoadMate.
  - Provides a list of the latest rides submitted to RoadMate. (Limit 5)
  - Provides any important news relating to RoadMate or its users.

- **Profile**
  The profile page serves two purposes:
  - Serves a read-only version of a users profile to visitors.
  - Serves an editable version of the profile page to the profile owner.

- **Offer Ride**
  The offer ride page provides all functionally pertaining to ride offer creation, modification and viewing.

- **Browse Rides**
  The browse rides page provide basic search functionally over the currently offered rides.
  The current search criteria are:
  - Source – Where the ride it leaving from.
  - Destination – Where the ride is going.

- **Support**
  Provides basic help information to visitors
  - Explains basic functionality of RoadMate.
  - Provides technical contact information.

- **About**
  Explains what RoadMate is, and what we're trying to achieve with it.

Future Prototypes

As part of our agile development process, the feature sets of future prototypes are not strictly defined, but instead hinge on what features remain after the first prototype has been delivered.

The following are features that are required as part of this project and will end up in future prototypes:

- OpenID
- Request Ride
- Ride Matching
- User Feedback
- Google Maps + Calendar

Additional features will be added as part of a brainstorming process after the delivery of the first prototype.

Software Quality Assurance

Unit Testing

As part of our software quality assurance process, we aim to make extensive use of unit testing to ensure our code meets both functional and non-functional requirements.

Selenium A suite of tools to automate web application testing

Selenium IDE is a Firefox add-on that records clicks, typing and other actions to make a test, which you can play back in the browser. Selenium also supports exporting your actions as unit tests, which can be integrated into your project.

With the help of the Selenium IDE we can generate unit tests, which emulate common actions users would make while using RoadMate. These site tests in combination with our server side tests give us complete coverage over the functionally provided by RoadMate.

Because these unit tests are platform agnostic, they easily integrate into our continuous integration process:

- **Developers**
  All unit tests must pass before code can be checked into the repository.
- **Local Server**
  Every 20 minutes our local server downloads the latest version of RoadMate from the repository. Once our site unit tests have been created, the local server will use them to determine whether to replace its working copy of RoadMate with the 'head' revision.

Usability

As part of our software quality assurance process, we aim for the highest standards in usability.

As a testament to this, our site functions without the need for JavaScript. We also have full intensions to meet government standards for accessibility to ensure the widest range of users have access our site.

Usability Testing

Usability testing is an intricate part of our development process and something we take very seriously.

In order to ensure our site meets the highest standards in usability (our own), we make use of internal and external parties to test our site and report issues back to us.

For external parties, we have the intension of using Silverback (see below) to record user experiences help us decide at meetings

Testing Matrix

As part of our software quality assurance process, we continually test our site against the major browsers for any inconsistencies.

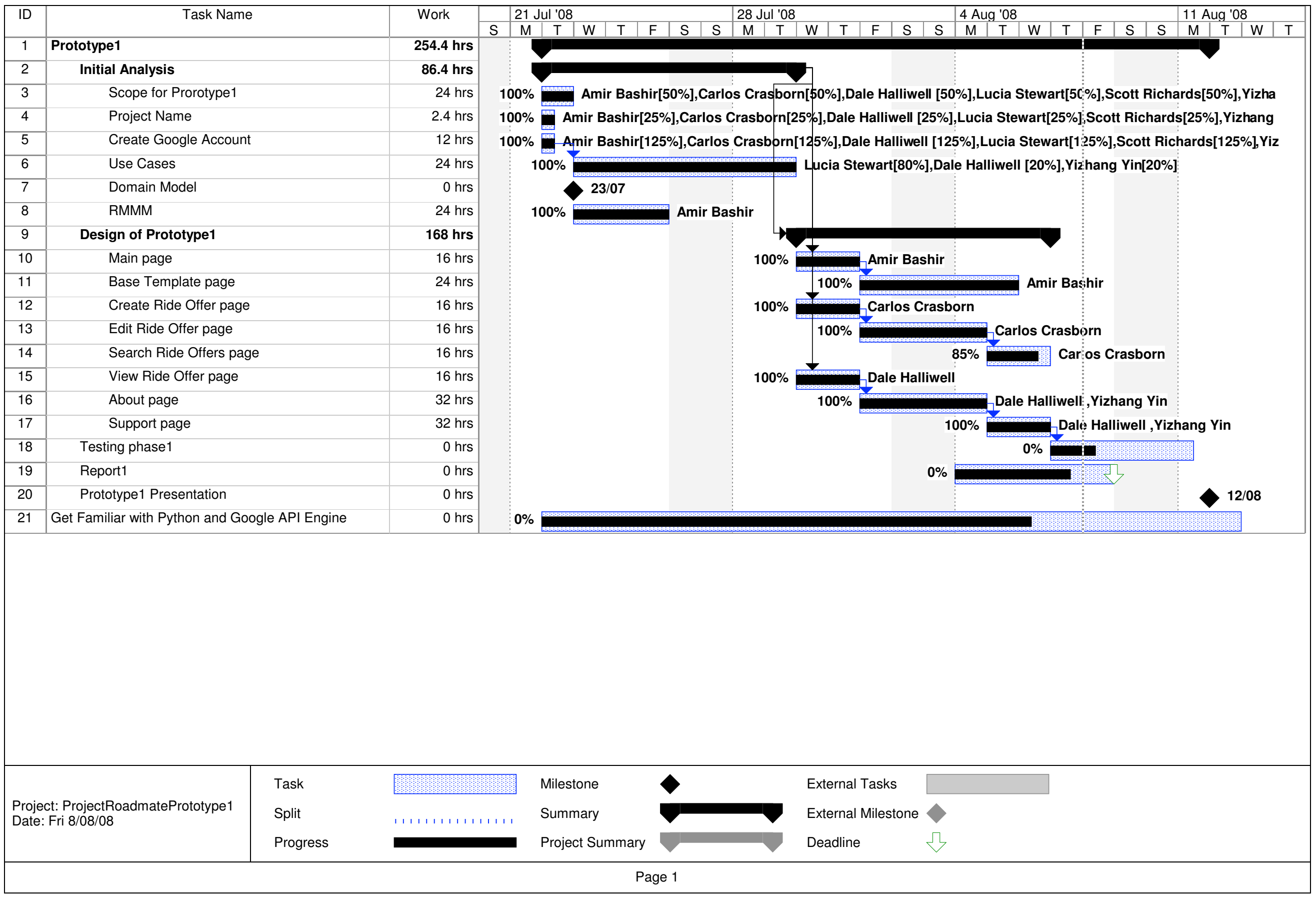
Testing is predominantly carried out on **Firefox** and **Internet Explorer 7,** as these are the browsers installed on our development machines. Testing on the reaming browsers is carried out when approaching a point release.

Silverback Spontaneous, unobtrusive usability testing software for designers and developers

Silverback makes it easy, quick and cheap for everyone to perform guerilla usability tests with no setup and no expense, using hardware already in your Mac. By capturing screen activity along with videoing the participant's reactions and voice, Silverback provides a complete usability-testing suite.

Gantt Chart

The Gantt chart below covers our project from inception through to delivery of our first prototype. The tasks following delivery of prototype one have been striped from this report, for aesthetic reasons. The full version can be downloaded from the repository (link).



ID	Task Name	Work
1	**Prototype1**	**254.4 hrs**
2	**Initial Analysis**	**86.4 hrs**
3	Scope for Prorotype1	24 hrs
4	Project Name	2.4 hrs
5	Create Google Account	12 hrs
6	Use Cases	24 hrs
7	Domain Model	0 hrs
8	RMMM	24 hrs
9	**Design of Prototype1**	**168 hrs**
10	Main page	16 hrs
11	Base Template page	24 hrs
12	Create Ride Offer page	16 hrs
13	Edit Ride Offer page	16 hrs
14	Search Ride Offers page	16 hrs
15	View Ride Offer page	16 hrs
16	About page	32 hrs
17	Support page	32 hrs
18	Testing phase1	0 hrs
19	Report1	0 hrs
20	Prototype1 Presentation	0 hrs
21	Get Familiar with Python and Google API Engine	0 hrs

Project: ProjectRoadmatePrototype1
Date: Fri 8/08/08

Task	▨	Milestone	◆	External Tasks	▬
Split	⋯	Summary	▬	External Milestone	◆
Progress	▬	Project Summary	▬	Deadline	⇩

Page 1

Individual Use Case Descriptions

Name:	**Register user**	
Actor:	**Primary: Visitor**	
Description:	Visitor to the site can register as Users	
Reference to Requirements:	The Visitor is redirected to Google Account Login Interface (will use an OpenID Registration Interface in later iterations)	
Preconditions:	None	
Related UC:	Login User	
Steps:	**Actor's action**	**System's action**
1	Visitor clicks on Register link	
2		System redirects the Visitor to the Google Accounts Login page where the user can login using a dummy test account
3	Login Selected	
4		System changes Visitors status to user and allows the user to view all pages.
Alternative Route		
3	Logout selected	
4		The Visitor's status remains as Visitor and the access to some pages is not allowed.
Postconditions:	If registration is successful, a new instance of User entity is created in the system (uses dummy user in this iteration)	
Assumption:		

Name:	**Login User**	
Actor:	**Primary: User, Secondary System**	
Description:	The User is redirected to the Google Accounts Login Interface (dummy test user can be used)	
Reference to Requirements:		
Preconditions:	None	
Related UC:	Register User	
Steps:	**Actor's action**	**System's action**
1		User is prompted to Login using available dummy test account
2	User selects Login option	
3		System changes users status to logged in and access to all pages and features is enabled
Alternative Route		
2	User selects Logout option	
3		System keeps user's status as Visitor and access to some pages and features stays disabled.
Postconditions:	An authenticated session for the User begins	
Assumption:		

Name:	Browse Rides	
Actor:	**Primary: User, Visitor Secondary: System**	
Description:	Visitors can browse rides by source and destination	
Reference to Requirements:		
Preconditions:	none	
Related UC:	None	
Steps:	**Actor's action**	**System's action**
1		User/Visitor is presented with the drop-down menus of source and destination towns
2	User/Visitor selects the source and destination and submits the query.	
3		All available matching rides are displayed. In this iteration when no rides are available, a blank screen appears.
Postconditions:	None	
Assumption:		

Name:	Offer Ride	
Actor:	**Primary: Ride Owner<<User>>, Secondary: System**	
Description:	A User creates a ride, specifying the relevant details, and the ride is submitted to the system for general display on the site.	
Reference to Requirements:		
Preconditions:	User is logged in	
Related UC:	None	
Steps:	**Actor's action**	**System's action**
1	User selects "offer ride"	
2		User is presented with a form where they specify the details of the ride they wish to offer
3	User enters the ride details into the form	
4	User clicks the save button	
5		System validates the form.
6		if all details are correct, the ride details are stored in the system and displayed.
Alternative Route		
5		if some details are incorrect, system notifies the Ride Owner and goes to step 2
Postconditions:	If successful, a new instance of Ride entity is created (with associated seats/passengers)	
Assumption:	None	

Name:	View Main, Support and About pages	
Actor:	**Primary: Visitor, User, Secondary: system**	
Description:	Visitor/User can view the pages on the RoadMate website	
Reference to Requirements:		
Preconditions:	None	
Related UC:	View My Profile/ My Rides page	
Steps:	**Actor's action**	**System's action**
1		Visitor/ User is initially presented with the main page that contains the links to other pages.
2	Selects the link to Support page	
3		The Support page is displayed in the browser window
Alternatjive Route		
2	Selects the link to About page	
3		The About page is displayed in the browser window
Postconditions:		
Assumption:	That trust is an issue significant enough to warrant developing a feedback system	

Name:	View My Profile/ My Rides page	
Actor:	**Primary: User, Secondary: system**	
Description:	A logged in user can view My Profile/ My Rides page	
Reference to Requirements:		
Preconditions:	The user must be logged in	
Related UC:	Edit My Profile	
Steps:	**Actor's action**	**System's action**
1	User selects a link to My profile page	
2		System verifies whether the user is logged in
3		If logged in My Profile/ My rides page is displayed.
Alternative Route		
3		If not logged in the User is redirected to the Login page
Postconditions:	User can access Edit link to edit My Profile details.	
Assumption:		

Name:	**Edit My Profile**	
Actor:	**Primary: User, Secondary System**	
Description:	User can change their own details (except for username). The User is presented with their personal details in editable form, they make the changes they wish, and submit the form. These details are validated and updated in the system.	
Reference to Requirements:		
Preconditions:	User must be logged in	
Related UC:		
Steps:	**Actor's action**	**System's action**
1		User is prompted to update their details in a editable form displaying their current details
2	User updates their details	
3		If critical details are invalid or incomplete, they are prompted to correct these, repeat step 2, else
4	User reviews and approves their details	
5		The User's details are updated in the system
Postconditions:	The changes to the instance of User entity are updated in the system.	
Assumption:		

Name:	**Edit My Rides**	
Actor:	**Primary: User, Secondary System**	
Description:	User can change their offered ride details. The User is presented with their existing ride details in editable form, they make the changes they wish, and submit the form. These details are validated and updated in the system.	
Reference to Requirements:		
Preconditions:	User must be logged in, user must have offered rides	
Related UC:		
Steps:	**Actor's action**	**System's action**
1		User is prompted to update their details in a editable form displaying their current details
2	User updates the ride details	
3		If critical details are invalid or incomplete, they are prompted to correct these, repeat step 2, else
4	User reviews and approves their details	
5		The User's details are updated in the system
Postconditions:	The changes to the instance of User entity are updated in the system.	
Assumption:		