CSCI 201L Final Project
15.0% of course grade

Introduction

Much work is occurring in the world of intelligent transportation. This work spans the spectrum of intelligent vehicles, vehicular vision, vehicular communication, smart cities, real-time traffic gathering, multi-modal flows, and many other areas. In the next decade, this work is only going to grow as vehicular technology continues to be at the forefront of consumer applications. In this project, you will get to simulate a few of these technologies to simulate the flow of vehicles through a transportation network.

This project will provide you with the experience of implementing a large project from conception to deployment. You will be required to submit project documents throughout the remainder of the semester following good software engineering principles. This project description document will provide you with some of the details that you will include in the high level specifications, though the specific requirements for your individual project will be determined by your group and approved by me. All guidelines included in this document must be implemented as stated unless approved otherwise.

Description

For this project, you will work in a group of 4-8 people. You are welcome to work with people from either section of CSCI201. Part of the purpose of this project is to work together in a team. There will most likely be some issues within the team, and I hope that you all will work out the issues. If you are unable, please let me know, and I will mediate a discussion. In extreme cases, I will dissolve groups or remove people from groups. At that point, the sub-groups will be responsible for the entire application.

This project will require you to create a graphical interface for a transportation network in LA. You will need to design the GUI portion of the application to ensure it includes everything that is required in this description. Your application will display the flow of traffic similar to http://maps.google.com and http://www.sigalert.com. The data you will display will be retrieved from a server running within the Viterbi School of Engineering. The location of that server and the format of the data will be provided to you.

The data you receive from the server will be from individual vehicles. These vehicles will provide you with their locations and speeds. You will need to aggregate all of the data you receive to determine the amount of time it takes to travel along a section of roadway. The speed reported from a vehicle only gives you a snapshot of the flow of traffic. To improve on this inaccuracy, you will use the proportional time algorithm to determine this flow. REFERENCE NEEDED

Instead of just showing the user a color on a section of the road based on the speeds, all of the vehicular data you receive from the server will need to be displayed on the map. Individual vehicles will be moving on the map based on their current speeds. The color of the vehicle will be shown based on the speed. The vehicle will move along the section of roadway based on that speed.

The data will be displayed on a mapping application, such as Google, Microsoft, or Yahoo Maps. You will need to find a mapping application to use, look up the API, and implement it in your application. Each map has different functionality and a different API, so make sure to research them sufficiently before deciding on one. This will be part of the analysis that goes into the design document.

Since there are already applications that display the flow of traffic, we are going to extend on that functionality. We will also provide the user a means of determining the fastest route to get from a specified starting location to a specified destination. You will provide the user with the amount of time it would take for him to travel to the destination at speed limit and at the current speeds.

All of the messages that you receive from the server will be stored in a database. This will allow fast retrievals for historical data. The database must implement SQL, and your program must connect to it using SQL statements.

The user will also be able to view historical data in an aggregated form to determine the times of the day that would be best for traveling from a source to a destination. You will display this data in a graph as well as a table. Exporting the data to formats such as comma-delimited will be required so users are able to import it into a spreadsheet, such as Excel.

All of the above specifications will be required for groups of four. If you have more than four people in your group, you will need to add requirements to the above. These will need to be included in the specifications document and approved by me as appropriate based on the number of people above four you have in your group. I suggest you talk to me outside of class to ask about specific requirements based on the number of people in your group before writing up specifications. Here are some additional topics that you could consider (though I encourage you to come up with your own):

- Vehicle-to-Vehicle communication
- Determining queue lengths or delay times at traffic signals or stop signs
- Identify alternate routes in real-time based on current delay on roads

Specific Requirements

You will need to abide by good object-oriented guidelines with your program. Part of your grade will be based on the efficiency of the design and implementation of your code as portrayed in your documentation as well as your code.

With multiple programmers on a team, you will need to set up a version control system for storing the programs. There are multiple version control systems available to use, including CVS, SVN, GitHub, etc.

You will submit documents for each major phase in the software engineering lifecycle – specifications, requirements, design, testing, implementation, and deployment. Your final program will be demonstrated during a specified time, and you will have to **prove** that your program is working correctly. We will not run specific test cases, though we may ask you to do something during the demo slot.

Tips

Since there are a lot of parts to the project, you may be tempted to assign specific parts to specific people. This is fine, but it is important that everyone in the group reviews each part before submission. You should probably have an internal deadline of at least one day earlier than the actual submission deadline so the rest of the group is able to review each document before submitting. Of course, with the implementation, this should be done must earlier so the entire group is able to test and work out bugs.

Much of the work might be able to be accomplished separately, especially with the technologies we have at our disposal for remote work. I am still going to require at least two hours a week to meet as a group and work out any issues and discuss upcoming tasks.

Since we are working on intelligent transportation applications in the project, there is a chance that you may come up with something that is publishable. If you think that you have come up with something that is novel, please talk with me and we can discuss trying to write a paper to submit to a conference or journal. This is not a requirement of the project, but it is definitely something that will help your resume for the future.

Submission Instructions

In Eclipse, your project should be named "CSCI201_Team#_FinalProject". For example, if you are assigned team number 2, you would name your project " CSCI201_Team2_FinalProject". This will help us for grading in case we need to run your program after submission.

You should always have a comment at the top of your code with your name in it. Do this at least once in the .java file where the main method exists. Include all of the team member names in this comment.

Use the "Export to archive file" functionality of Eclipse to submit a .zip file of your project. This is summarized at:
http://agile.csc.ncsu.edu/SEMaterials/tutorials/import_export/

Once you export it, try importing it yourself using the instructions on the same page to make sure everything will work as intended when the grader imports it using the same methodology.

Include a README.txt file in your project in the doc/ folder of your project that explains what works and what doesn't, along with any other nuances with your project about which the teaching staff should be aware.

Grading Criteria

% of Final Grade	**Due Date**	**Criteria**
1.0%	2014-03-09	Submit team members and weekly meeting time
1.0%	2014-03-16	Specifications document
1.0%	2014-03-30	Requirements document
2.0%	2014-04-06	Detailed design document
1.0%	2014-04-13	Testing document and testing code
1.0%	2014-04-20	Code completion (though testing and bug fixes can still be implemented)
1.0%	2014-04-27	Deployment document
5.0%	2014-04-30/ 2014-05-01	Demonstration/Presentation
2.0%	2014-04-30/ 2014-05-01	Complete documentation

Description of Grading Criteria

All documents should be submitted as PDFs.

Submit team members and weekly meeting time
You will need to select your group of between 4 and 8. You also need to find a two hour block where the entire team is able to meet each week. Determine a location to meet. Submit a document that includes all of the above information.

Specifications document
All teams will need to write up all of the details that I have included in this document in a specifications document. If you have more than four people in your group, there will be additional specifications that you need to include. Clearly delineate the additional specifications so I know what you will be implementing in addition to the provide documentation.

Your specifications should not be copied from this document. I have just provided you with a project concept document. You need to then take this and write out specifications as if you were the client requesting for a program to be written that performed the tasks provided.

Requirements document
Based on the feedback you receive on your specifications document, you will then need to write up technical requirements. You will submit both your specifications document and your detailed requirements document since you may have had to modify your specifications based on provided feedback. Make sure you clearly delineate anything that you added, deleted, or changed in your specifications. This can be accomplished using Track Changes in Microsoft Word (or a similar feature in other word processors).

Detailed design document
Based on the feedback you receive on your requirements document, you will write up your detailed design document. This should show any difficult algorithms, the exact GUI (though slight modifications will still be allowed), the database schema (ER diagram), the hardware/software requirements, class diagrams, inheritance hierarchies, etc. This will likely be a lengthy document.

In addition, submit your specifications and requirements documents again with any changes clearly delineated using a feature such as Track Changes.

Testing document and testing code
You will need to prove to us that your program is working the way it is supposed to. That means that you will need to include sufficient test cases to prove that your program is working. This includes all types of testing, including white box testing, black box testing, unit testing, regression testing, etc. Not only should you create a document in this step, but you should also create the test code.

You will be able to add other test cases and code throughout the implementation phase, but this will hopefully provide you with a good set of test cases to ensure proper operation of your application.

In addition to the testing document, submit your specifications, requirements, and detailed design document with any changes clearly delineated using a feature such as Track Changes.

Code completion (though testing and bug fixes can still be implemented)
You will submit your code wrapped up in a zip file that can be imported into Eclipse and executed. By this date, the new development of your program should be completed, and only bug fixes and testing should be performed. You shouldn't be adding new features into your program after this date.

In addition to the code, submit your specifications, requirements, detailed design document, and testing document/code with any changes clearly delineated using a feature such as Track Changes. You probably will have added more test cases as development proceeds, so mark those.

Deployment document
We won't actually be deploying your program in a live environment, but if we were to, this document should be what is followed. It needs to include step by step of what to do to take the code from Eclipse and deploy it on a server. This will most likely consist of some other configurations also (i.e. outside libraries, server settings, Java download/install, etc.).

In addition to the deployment document, submit your specifications, requirements, detailed design document, testing document/code, and updated code with any changes clearly delineated using a feature such as Track Changes. You probably will have added more test cases and fixed some bugs as development proceeds, so mark those.

Demonstration/Presentation
You will sign up for a demonstration slot. You will demo your program and prove to us the correctness of it. We may ask you to show something specific, but you will be driving the demonstration. No formal slides or presentation is required, but you will have to walk us through the program and show us all of the features. I assume this will take between 5-10 minutes.

Nothing is needed to be submitted for this part.

Complete documentation
On the demonstration day, you will provide complete documentation. You will put all of your documents together into one document, include a cover page, table of contents, and page numbers, and submit the entire set of documentation. Again, if anything has changed from the last submission, mark the changes clearly with a feature such as Track Changes. This document should be nice enough that it could be printed and presented to the client of this application as the final document.