ECE 297 – Design and Communication

Course Syllabus, January 2017

Lecturers and Office Hours:

	Design	**Communication**
Professor	Vaughn Betz	Ken Tallman
Office Location	311 Engineering Annex	Sanford Fleming, B670 (Engineering Communication Program Offices)
Email	vaughn@eecg.utoronto.ca	k.tallman@utoronto.ca
Office Hours	Any time you find me in my office, or email to book an appointment	By appointment

"Design is not just what it looks like and feels like. Design is how it works."
- Steve Jobs

"The art of communication is the language of leadership."
- James Humes (Presidential speechwriter & author of Apollo 11 plaque left on the moon)

Course Overview: This course involves designing and completing a large software project in a team, and communicating effectively with both technical supervisors and client-facing business managers. The course focuses on practical skills in all these areas -- design, software development, oral communication and written communication – as these skills are essential to having a successful engineering career. The evaluation is similarly practical: your grade will be based on the quality of your software design, and the calibre of your oral and written communication.

The information concerning the course project and evaluation is subject to change if circumstances warrant; such changes will communicated to students as early as possible if they are necessary.

Required Text: **Made to Stick: Why Some Ideas Survive and Others Die**, Chip and Dan Heath, Random House, 2007. Available in the U of Toronto bookstore, or you can buy it for ~$20 from Amazon.ca.

Course Web Page: http://www.eecg.toronto.edu/~vaughn/ece297

Lecture Notes: We will post any slides we present on the course website after the lecture.

Prerequisites: ECE 244 (Programming Fundamentals) or equivalent. The course project is completed in C++ so you must be familiar with this programming language.

Lectures: Monday, 9 am – 10 am, MC 102
Friday, 11 am – noon, MC 102

Key information on programming, algorithms and communication will be presented during lectures; you will need this information to complete the course project.

Tutorials: Three time slots; depending on your section you will attend one time slot:
Monday 1 pm – 3 pm (GB 304 / GB 404)
Wednesday 9 am – 11 am (BA 1180)
Thursday 3 pm – 5 pm (SF 3201 / GB 304)

From January 9 to 20, (the first two weeks of tutorials) important programming information on how to use the C++ STL library that will be essential to completing your project will be taught in the tutorials. Accordingly, you should consider the first two weeks of tutorials *mandatory* and attend the entire two-hour tutorial time slot for your section. The week of Jan. 23 you will meet your Communication Instructor (CI) during the tutorial and choose a weekly meeting time; from Jan. 30 onwards your team of three will meet with your Communication Instructor during that agreed-upon time.

During some weeks from Jan. 23 onwards, there will be *optional* one hour coding tutorials that give additional guidance on good software architecture. The topic of each of these tutorials will be announced in advance, and the tutorial will be given twice in each two-hour tutorial period, so you can attend even if your regular meeting with your CI is scheduled during your tutorial time period.

Labs: Three time slots; depending on your section you will attend one time slot:
Monday 10 am - noon (SF 2102 / GB 251 / GB 243 / SF 2204)
Monday 1 – 3 pm (SF 2102 / GB 251 / GB 243 / SF 2204)
Monday 4 - 6 pm (SF 2102 / GB 251 / GB 243 / SF 2204)

Labs begin immediately, on Jan. 9. For the first two weeks you will work individually in the labs to learn and demonstrate the software tools we use this term. From January 23 onwards, you will work in teams of three in the lab and will be assigned a specific TA to mentor & grade your team; you must meet weekly with this TA.

Project and Organization:

In this course you will build a software program that will provide functionality similar to Google Maps, plus some other features. By the end of the course your program will be able to

- Read in a database of all the intersections and streets in a city.
- Draw the resulting map nicely and let the user interact (pan, zoom, highlight, search for locations, etc.) with it.
- Find travel routes between two intersections in the city and give directions to a user.
- Find a good order of deliveries and a good driving path for a courier company driver to complete his/her list of daily deliveries.

You will be evaluated by two people in this course.

1. A **Teaching Assistant (TA)**, who is a graduate student in engineering and will mentor you on design and software development, and evaluate the quality of your software project.

2. A **Communication Instructor (CI)**, who is trained in effective communication and will mentor you on oral and written communication, and who will evaluate your written documents and the majority of your oral presentations.

You will **work in teams of three students, and you will choose your teammates**. *All three team members must be in the same time slot for labs and for tutorials (there are 3 slots)*, to make scheduling meetings with the teaching assistants and communication instructors feasible. You must choose your team members during the first two weeks of the course. Each team will then be assigned one CI and one TA, who will mentor you and grade you for the duration of the course.

From the week of January 23 onward, you will meet with your TA during your lab period to update him/her on your status and sometimes to demonstrate your progress toward a project milestone. You will also meet with your CI in another meeting to update him/her on your progress in creating presentations and documents and to get feedback on these deliverables and potentially on the user interface of your software design. Your team will also have a wiki page (an easy-to-edit web page), and you should always update your wiki page before your status meetings to highlight what you have achieved each week, any challenges you have encountered, and what the steps for the next week are.

Detailed Deliverables and Evaluation:

There is no midterm or final in this course; your mark will be determined entirely by the quality of your design and software, your oral and written communication, and how well and professionally you communicate your status throughout the course. The detailed list of deliverables and the mark distribution are given below. Full details on the requirements for each software or communication deliverable will be given in separate documents distributed when that deliverable is assigned later in the course.

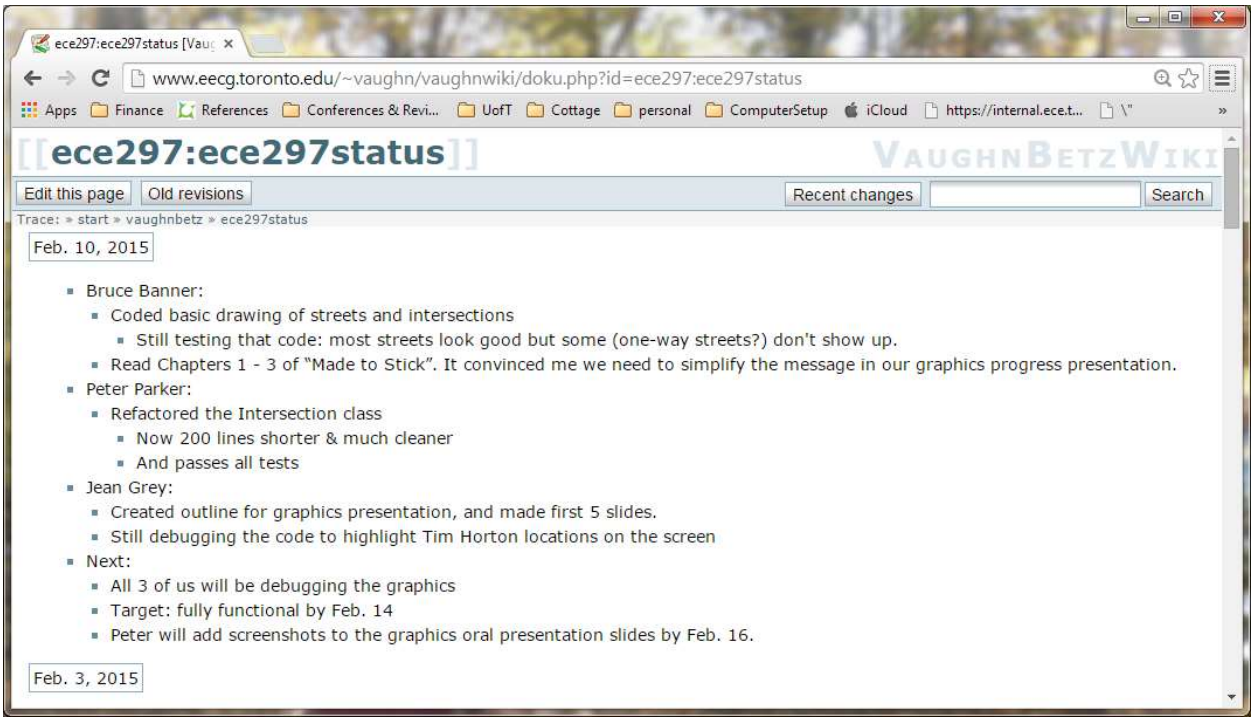
Deliverable	Marks	Due Date	Marked by
Milestone 0 - basic SW tools & debugging (individual)	2	Monday, Jan. 23	**TA and automarker.** You will debug a program we give you, submit the fixed code & demo that you can use source code control and debugging tools to the TA.
Milestone 1 – create & load data structures, answer simple queries	9	Monday, Feb. 6	**TA and automarker.** Submit code. The code will be graded for correctness and style.
Code review	2	Monday, Feb. 13	**TA**. You will be given another team's Milestone 1 code, and will write a brief document detailing what you like and don't like about the code style. The TA will mark the quality of your feedback.
Milestone 2 - graphics: draw map	13	Monday, Feb. 27	**TA.** Submit code & demo to TA. Your mark will be based on the quality of your map visualization, and on the code style.
Milestone 3 – finding travel routes between points	13	Tuesday, March 21	**TA and automarker.** Submit code & demo to TA. You will be marked on the performance of your path-finding algorithm, the style of your code, and the usability of travel directions and user interface.
Milestone 4 – find an order and route for a set of courier deliveries to minimize travel time	11	Thursday, April 6	**TA and automarker**. Submit code. You will be marked on the performance of your route-finding algorithm.
Weekly status reporting to TA	3	All Semester	**TA.** Write status reports on wiki each week. Arrive at lab prepared to show code status, and summarize progress and next steps well.
Weekly status reporting to CI	3	All Semester	**CI**. Write status reports on wiki each week. Arrive at tutorial prepared to show document and project status, and summarize progress and next steps well.
Written Document 0: Presentation Analysis Using Made to Stick (Individual)	3	Monday, January 30	**CI**. A brief written analysis of the video of a Google I/O presentation, highlighting how the presentation used Made to Stick principles and areas where it could be further improved.
Written Document 1: Graphics Proposal	10	Friday, Feb. 17	**CI**. This is a written document summarizing your background research into how Geographic Information Services (GIS) applications are made easy-to-use, and a plan for how you will make your map visualization

			program usable.
Oral Presentation 1: graphics functionality / usability	10	March 13 – 16, in your tutorial timeslot	**CI**. Oral progress presentation detailing your original plan for visualizing the map, your evaluation of the implementation and any changes that you made.
Written Document 2: lessons learned (individual)	3	Thursday, April 13	**CI**. A short individual report where you reflect on any issues you encountered in teamwork, project management or communication, and give some concrete steps you would take on future projects to mitigate these issues.
Oral Presentation 2: project summary & pitch	18	During Exam Period (April 17 or later)	**CI and TA**. This is a final presentation summarizing what you achieved in your program, and it is intended for both a client and engineering management audience. You should describe the interesting features of your program, including screenshots (and possibly a brief demo), and you should give a "pitch" for a future feature(s) that you could add to the program to address some business opportunity; you want to convince the clients (CI and TA) to fund development of these new feature(s).

All deliverables except those shaded and marked as "individual" will be completed in a team of three students. Note that while you will complete most deliverables as a team, the marks of individual team members can and will be different in some cases, based on the relative contribution of each team member and the quality and scope of the portion of the design / document / presentation created by each team member.

Submission deadlines will be at **6:00 pm** of the date in question. When a demo is required it will be done during your lab period in the week of the submission deadline. *The milestone (software code) submission deadlines are firm as completing a project on time is a key part of project management*. The electronic submission procedure will be disabled at the due date and no late submissions will be accepted except in exceptional circumstances. You should plan to complete your software well before the due date to avoid last minute submission problems.

You are expected to meet each week with both your TA and CI, and have a written (on your wiki page) status update. If you cannot be present some week for some reason, it is up to you to make alternative arrangements with your TA and/or CI in advance. The screen shot below gives an idea of what a short status report might look like. Your status reports should list not just what the team has done and will do next, but should also give specific information on what *each team member* did or did not achieve in the past week, and what *each team member* commits to complete in the coming week. Documenting your plan and your commitments to each other on the wiki is an important part of project management.



There will be prizes (bonus marks and small gifts) for the teams that achieve the best solutions to Milestone 4.

Note that the new feature(s) you propose in your final presentation will not be implemented during the course. You are making a *pitch* for funding to the client, so the key is to show an interesting idea and a compelling case of why it makes sense to build; you are completely unconstrained by any need to actually build it!

Suggestions and Workload:

To spread the workload of this course, we begin the labs and tutorials immediately (Jan. 9). Get started on Milestone 0 and Written Document 0 right away! Your software deliverables will depend on knowing the tools that you are taught in milestone 0, so make sure you go through all the tutorials and complete Milestone 0 well. To complete Written Document 0 you will need to read the first six chapters of **Made to Stick**. You will also be expected to apply principles from **Made to Stick** in your documents and presentations, and be able to answer questions on how you applied the principles. For all these reasons you should read **Made to Stick** as early as possible, and certainly within the first three weeks of the course.

Later software milestones will depend on the code you write for earlier milestones, so it is important you do not fall behind on the early milestones, and that you create high-quality, maintainable software so you can keep extending it through the course.

By Friday, Jan. 20 you will need to have formed a group of three, with all three teammates being in the same time slot (for both tutorial and labs – there are three different time slots). Get started on finding teammates right away, and be strategic in your choices. You want to choose teammates that:

- Have similar work ethics and goals for project quality / grades;

- Have complementary skill sets (e.g. a good writer plus a good coder plus a good presenter);
- Get along and work together well on a personal level.

You will enter your team composition from the Team Selection System on the course webpage. Balancing work well across the three team members (and having a solid contribution from each) is key to doing well in this course with a reasonable workload.

Lecture Material:

The lecture material will help you complete the course milestones, documents and presentations. Additionally, you will be asked questions after your presentations, and any material covered in the lectures and any material in the first six chapters of **Made to Stick** will be assumed background knowledge on which you should be able to answer questions.

The material presented in lectures and during the (mandatory) first two weeks of tutorials will include:

First two weeks of tutorials:

- C++ templates
- The C++ Standard Template Library classes, including vectors, lists (linked lists), maps (binary trees) and iterators.

Lectures:

- Course goals and organization
- Source code control and Subversion (Milestone 0)
- Made to Stick and effective writing overview (Written Document 0)
- Team dynamics and project management basics
- Graphs and data structures to represent them (Milestone 1)
- Good software style
- Testing, unit tests and software profiling and coverage tools
- Introduction to computer graphics (Milestone 2)
- Background research and writing a clear summary and plan (Written Document 1)
- Effective oral communication (Oral Presentation 1)
- Finding paths/routes in graphs (Milestone 3)
- The traveling salesman problem (Milestone 4)
- Integrating text and graphics in communications
- User-focused communication
- Handling questions well
- How to pitch a technical business idea (Oral Presentation 2)

Laboratory Computers and Working from Home:

In addition to the computers in your lab room, you may also use your home computer to work on assignments. You may do so by remotely accessing the lab machines (e.g. ug150.eecg.utoronto.ca) from home through your Internet Service Provider (ISP). To do so, you must connect to ECE using a secure shell (**ssh**) and if you want to display graphics, through the Virtual Network Computing (**VNC**) program. For more information on using the ECE lab computers remotely, please consult the **VNC Quick Start Guide** on the course website.

You may also use your home computer by downloading an *experimental* virtual machine from the course website; this will make the tools you require such as NetBeans and various libraries available

when you run the virtual machine on your home computer. Some tools, such as the program that actually submits your code for marking will work only from the ECE machines (which you could have logged into remotely using vnc). If you decide to install a virtual machine and work on your project on your home computer, **you must still ensure your code works correctly on the lab ECE computers. A program that does not work correctly on ECE, even if it works correctly on your home machine, will be marked as incorrect!** Plan ahead, and give yourself the time to test what you developed at home on ECE before the deadline.

Getting Help via the Discussion Board:

If you have a question outside of your regular lab (TA) or tutorial (CI) time, you can post it to the course discussion board; we will use piazza for the discussion board and there is a link to it on the course website. The TAs as well as the instructors will regularly check this board to answer questions and the answers become available to all students. Thus, it pays to check the board before posting a question to make sure that the question has not been posted and answered earlier.

You are encouraged to use the discussion board, and to engage in discussions about the assignments with fellow students on the board. **However, do not post code on the discussion board** or give detailed solutions or algorithms on the discussion board. Doing so will be treated as an academic offense (see below). Also, if the message is to be directed to a specific TA, CI or instructor, use email instead of the discussion board.

Independent Work and Academic Integrity:

Students from different teams are encouraged to discuss with one another issues and problems that arise in the course of completing the design project. Such discussions often provide interesting contrasts in design choices and can be very valuable learning experiences.

However, **work submitted for credit must be the student/team's own work**. It is one thing to discuss and compare, but quite another to rely on some other team's work to obtain credit for an assignment. It is also an offence to knowingly allow a copy of your work to be submitted by another person/team for credit. It is also an offense not to put in place protections to prevent your code from being copied without your knowledge.

A reasonable rule of thumb to follow during a discussion that crosses teams is that nobody leaves the discussion with written notes of what was said. It is also unwise to detail the entire algorithm or set of data structure choices that one team is making to another team. It is unlikely that two teams who have discussed only high-level approaches to a problem with no written notes will write highly similar programs.

All programs submitted for credit in this course will be compared pair-wise to identify cases of collusion, copying, and similar offenses. A sophisticated program that is capable of detecting similar programs even if considerable effort has been taken to conceal their similarity does the comparison. Use of basic software libraries such as STL is fine in your code, but use of significant amounts of code you find from others (e.g. on the web) to implement a milestone is not allowed. If in doubt about whether you can use code available in a library or on the web ask your TA or post a question to the instructors on piazza.

All written reports and oral presentations must similarly be the work of a single team and will be submitted to **turnitin.com** for comparison to the documents of other groups and to published works for plagiarism detection.

Any work submitted for credit that is not the work of the person submitting it will be treated as an offense under the Code of Academic Discipline of the University. Similarly, aiding anyone in the submission of work that is not the work of the submitter will also be treated as an offense under the Code of Academic Discipline of the University. The Code of Academic Discipline will be rigidly enforced in this course. Penalties can range from grade penalties in the course to suspension from the University. The Dean, as advised by the instructor and the Departmental Chair, determines the penalty for each case.

Normally, students will be required to submit their course essays to Turnitin.com for a review of textual similarity and detection of possible plagiarism. In doing so, students will allow their essays to be included as source documents in the Turnitin.com reference database, where they will be used solely for the purpose of detecting plagiarism. The terms that apply to the University's use of the Turnitin.com service are described on the Turnitin.com web site.