

# Welcome to CS310

2014, Summer Term 1

Instructor: **Elisa Baniassad** ([ebani@cs.ubc.ca](mailto:ebani@cs.ubc.ca), room 369)

*Thanks so much to Meghan Allen, Eric Wohlstadter, Gail Murphy,  
and a lot of other people for the slides/course design/etc!!*

# Let us consider, the Ariane-5

[http://youtu.be/gp\\_D8r-2hwk](http://youtu.be/gp_D8r-2hwk)

Ref: ARIANE 5 Flight 501 Failure Report by the Inquiry Board



[http://www2.vuw.ac.nz/staff/stephen\\_marshall/SE/Failures/SE\\_Ariane.html](http://www2.vuw.ac.nz/staff/stephen_marshall/SE/Failures/SE_Ariane.html)

Sadly, the primary cause was found to be a **piece of software which had been retained from the previous launchers systems and which was not required during the flight of Ariane 5**. The software was used in the Inertial Reference System (SRI) to calculate the attitude of the launcher. In Ariane 4, this software was allowed to continue functioning during the first 50 seconds of flight as it could otherwise delay launching if the countdown was halted for any other reason, this was not necessary for Ariane 5. As well, the software contained implicit assumptions about the parameters, in particular the horizontal velocity that were safe for Ariane 4 but not Ariane 5.

**The failure occurred because the horizontal velocity exceeded the maximum value for a 16 bit unsigned integer when it was converted from it's signed 64 bit representation. This failure generated an exception in the code which was not caught and thus propagated up through the processor and ultimately caused the SRI to fail.** The failure triggered the automatic fail-over to the backup SRI which had already failed for the same reason. This combined failure was then communicated to the main computer responsible for controlling the jets of the rocket, however, this information was misinterpreted as valid commands. As a result of the invalid commands, the engine nozzles were swung to an extreme position and the launcher was destroyed shortly afterwards.

**The failure was thus entirely due to a single line of code.**

# Bad SE practices create...

- Failed projects
- Lost money
- Stressed employees
- Poor customer value

*To our customers,*

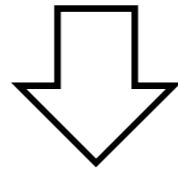
*At Apple, we strive to make world-class products that deliver the best experience possible to our customers. With the launch of our new Maps last week, we fell short on this commitment. We are extremely sorry for the frustration this has caused our customers and we are doing everything we can to make Maps better.*

# What is Software Engineering?

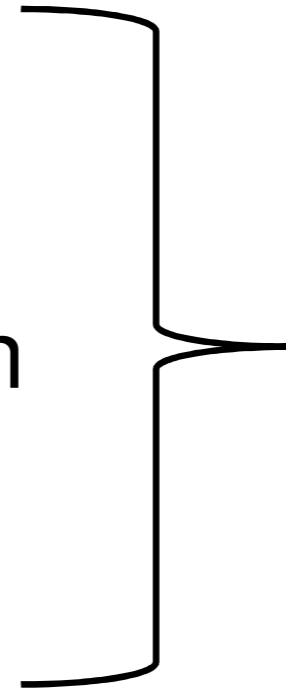
People working **together** to create a **robust** software system that satisfies the **client**.

# The Phases of Software Engineering

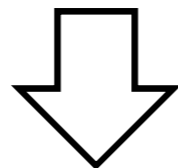
**client**



1. Requirements
2. Design
3. Implementation
4. Testing
5. Maintenance



Process  
ties  
**together**



**robust** software system

# SE Here, versus SE in the world

	Class project	Industry project
Time	1-2 weeks	Years, months
People	1-2	10 - 100s
LOC	100s	Millions
Capital at stake	0	Millions

# Quick introductions!

- My name is Elisa Baniassad ([ebani@cs.ubc.ca](mailto:ebani@cs.ubc.ca))
- ICICS 369
- Office hours — probably in here (or in my office) after each class for 45 minutes. If you need to see me individually, please email me for an appointment.

# About you!

- Please fill in an index card for me so I can get to know you!
- Please include:
  - your name (as listed in the registration system)
  - your preferred first name
  - something interesting about yourself to help me remember you
  - why you are taking this course
  - what you'd like to learn in this course



# Announcements

- **LABS:**

- **Start this week!** They run Wednesday **AND** Friday (yes, you must attend two labs/week).
- **Attendance is absolutely mandatory.** Lots of times assignments/deliverables are handed in right in lab. Marks will likely be returned in lab. In addition, you will lose 1% of your overall grade for each lab you miss without an acceptable reason. If you have to miss a lab, please tell your team and your TA before the lab begins, and provide sufficient documentation.

# Communication

- **Ask course related questions on Piazza.**
- You can choose to make these public if they are questions you're willing to ask "loudly".
- Otherwise, you can PM your lab TA or me, if it's a question about the course generally.

# Resources

- Course **web page**: <http://www.ugrad.cs.ubc.ca/~cs310>
  - the website has a calendar, and all the lectures. It also has all the instructions for the project deliverables.
- **Grades**: I will post information about how we will return grades to you (potentially on paper, in class or lab, potentially through connect)
- **Piazza** for discussions: <http://piazza.com/ubc.ca/summer2014/cpsc310> **(SIGN UP RIGHT AWAY!!!)**

# The Project

- Large-scale development project
  - Simulates “real life” challenges
  - Uses tools/toolkits
  - Necessitates team work, following a Scrum software team process
  - 4 students per team (form groups in your lab)
  - TAs will work with you, and take a Scrum-master/  
Facilitator role in your team

# Project is Continuously Evaluated

- Project goes in phases (called “assignments”)
  - **1, 2:** Learn tools and toolkits
  - **3:** Elicit requirements/create product backlog
  - **4:** Create your initial design
  - **5:** Plan your sprints
  - **6:** Implementation sprint (ends with a demo)
  - **7:** Final sprint (s/w release candidate) (ends with a demo)
- In phases 3-7 you will have “daily scrum” standup meetings in your lab, for 5-10 minutes with your TA. This will help you discuss what you’ve done, and plan your next moves.

# Grading

- 35% Project/Assignments
- 40% Final
- 20% Midterm
- 5% Lecture participation
- **PLEASE NOTE:**
  - **you must pass the final, project, and sprints to pass the course!**
  - your daily scrums will be part of your project grade
  - peer evaluations will form part of the grading scheme for your project

# 5% Participation grade

- I will be taking attendance (I KNOW, I KNOW) and will come around during class activities and see that you're involved.

# Course Schedule Overview








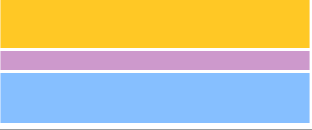
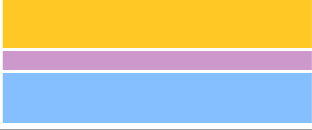









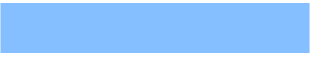
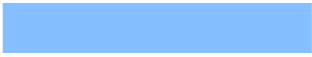


- **A single (pre-exam) week looks like this:**

MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
<p>9:30-12 Lecture</p>		<p>9:30-12 Lecture</p>		<p>9:30-12 Lecture</p>
<p>12-12:45 Office hours</p>		<p>12-12:45 Office hours</p>		<p>12-12:45 Office hours</p>
		<p>Later that day... LAB</p>		<p>Later that day... LAB</p>



# Course Schedule Overview

- The whole course looks like this:

week #	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
1: May 12-16					
2: May 19-23					
3: May 26-30					
4: June 2-6					
5: June 9-13					
6: June 16-20					
June 23-27	<b>EXAM PERIOD</b>				
7: June 30-July 4					
8: July 7-11					

Lecture
Office hours
Lab

# Do Not: Copy/Plagiarize!!!

- Borrowing is a big part of software development
- However this class is about building **your skills**
- **DO:**
  - make use of APIs to help your project
  - collaborate where specified
  - work individually otherwise
  - give credit if you got “inspired” by something you saw on line.
- **DO NOT:**
  - copy code wholesale from an uncredited source
  - think we won't be able to tell!!!

# Keys to success

- **Attend lectures** and labs (all of them — seriously)
- **Stay up to date** on readings
- **Do your share of the work** in your team (your peer reviews will affect your grade - everyone should try to contribute equally)
- **Keep up with the project milestones**, and start longer milestones early so you're not pressed at the end of each phase.
- **Stay courageous** and energetic with the tools (I know — they're a pain, that's just how SE is)...

Because... what will you really be doing?

The naked truth about SE  
is that it is mainly about:

**FIGHTING WITH TOOLS!!!!!!**

# After CPSC310 you will be able to...

- Explain the **technical** and **interpersonal** challenges of software development
- **Communicate** technical matters with programmers, managers, and clients effectively
- Perform the **activities/phases** of software development effectively using modern **methodologies** and **tools**.

*And you'll have more confidence fighting with tools!!*