- **CSCI 361**
- Lab and Lecture sessions
  - Lectures for framing ideas and concepts
  - Labs devoted to exercises, practical examples, and live grading
- Course run by Prof. Sterling and Prof. Boranbayev
- *Teaching Assistant*: Asset Berdibek, Adil Sarsenov
- *Contact Info*: Mark Sterling, Office 7E440

Objectives for today

- Software Engineering: define what it is and provide some motivation
- Review the aims and management of the course

The Software Crisis[1]

- Greater computing power ⇒ Challenges for software development
- The *Software Crisis* 1968
  - Capability of software lags behind that of the hardware
  - General difficulty of developing complex software systems

[1] pictured: IBM System/360

Our moment in computing: new and old

NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

- The idea of mechanical assistance to speed calculations goes back to antiquity
  - For example, what is the meaning, in english, of the word *pebble*
- Continued into the industrial revolution: Hollerith machine, Jacquard Loom, Babbage's *Analytical Engine*
- In the 20th century, WWII drove research in computing
  - Cryptanalysis
  - Anti-aircraft
- Innovations like solid-state electronics, integration, the *stored program* computer

"...considered harmful"

NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

- 1968 NATO Conference: first use of the term "Software Engineering"
  - At the time military was the main client for software[2]
  - Frustration with projects being delivered late and over budget
- The first software crisis led directly to many of the innovations that we take for granted today
  - *Structured Programming*[3]: idea that you should only use certain kinds of flow control (no *jump* or *goto* statements)
  - High-level languages: for example, *Simula* (1967) was the first object-oriented language

---

[2] Pete McBreen. *Software craftsmanship: The new imperative*. Addison-Wesley Professional, 2002.

[3] Edsger W Dijkstra. "Letters to the editor: go to statement considered harmful". In: *Communications of the ACM* 11.3 (1968), pp. 147–148.

The Software Crisis Today

- The software crisis is a *perennial* problem
- The CHAOS report published by the Standish group[4] surveys the reasons software projects fail provides a more current view of the crisis
- *What is failure (for software projects)?*: cancellation, over-budget, final product lacks a requested feature
- *What are the reasons for failure?*[5]
  - Incomplete or changing requirements
  - Lack of user input and involvement
- Let's review some examples of software failures

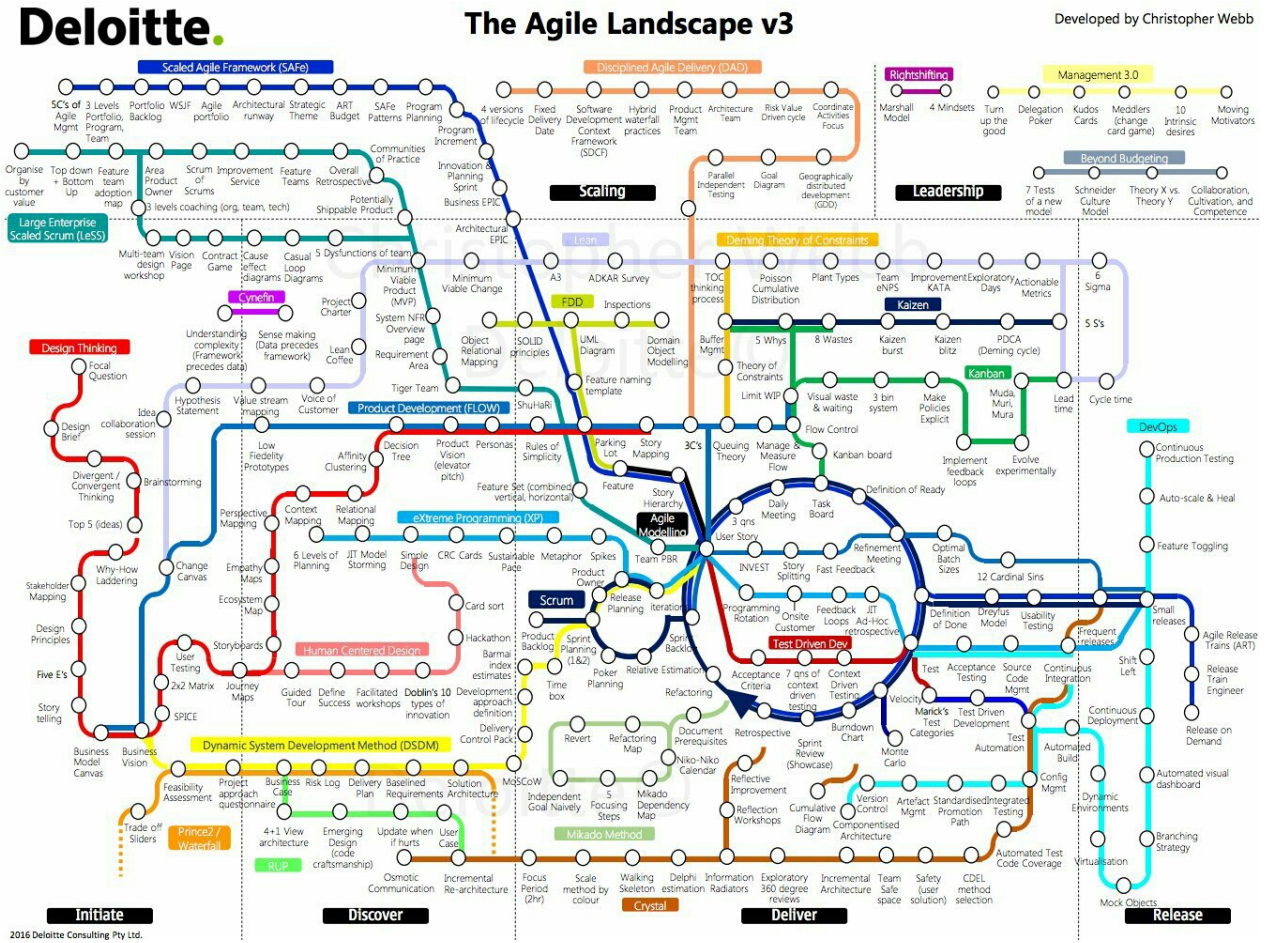
[4] Khaled El Emam and A Günes Koru. "A replicated survey of IT software project failures". In: *IEEE software* 25.5 (2008).

[5] Frank Tsui, Orlando Karam, and Barbara Bernal. *Essentials of software engineering*. Jones & Bartlett Learning, 2016.

The Software Crisis Today (cont.)

- We continue to witness the failure of high-profile software projects
- Nearly all countries and industries fundamentally rely on computer systems
  - Spheres affected: government, science, industry/consumer markets, open source community
- Human life is increasingly dominated by technology (e.g. Social Media, "Big Data" technologies) and the trend seems unlikely to reverse
- In "Mission Critical" systems failure can be catastrophic (e.g. aviation or the power industry)

The Agile Landscape[6]



- Agile Methods visualized as a map

[6] C. Webb, Deloitte, http://blog.deloitte.com.au/navigating-the-agile-landscape/

Failure in Government: healthcare.gov (2013)[7]

- Obtaining health insurance "like ordering a pizza"?
- *Consequence*: nearly derailed a signature piece of legislation, created unnecessary financial uncertainty for millions of Americans

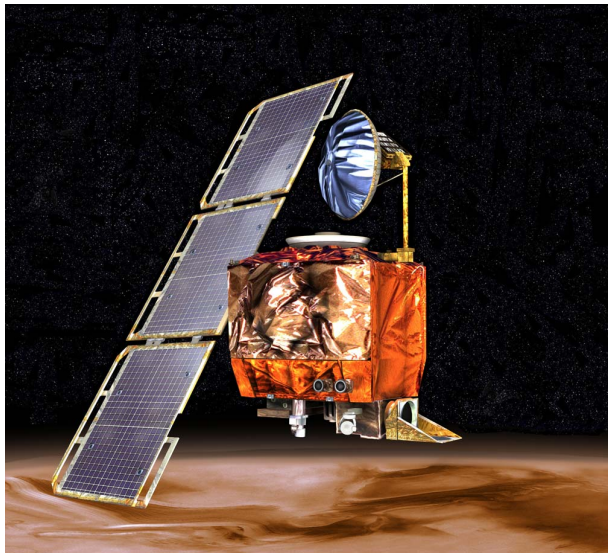
[7] Jane Cleland-Huang. "Don't Fire the Architect! Where Were the Requirements?" In: *IEEE software* 31.2 (2014), pp. 27–29.

“A series of tubes”

Sen. Stevens’ comments on the Internet

Ten movies streaming across that, that Internet, and what happens to your own personal Internet? I just the other day got an Internet was sent by my staff at 10 o’clock in the morning on Friday. I got it yesterday [Tuesday]. Why? Because it got tangled up with all these things going on the Internet commercially.
They want to deliver vast amounts of information over the Internet. And again, the Internet is not something that you just dump something on. It’s not a big truck. It’s a series of tubes. And if you don’t understand, those tubes can be filled and if they are filled, when you put your message in, it gets in line and it’s going to be delayed by anyone that puts into that tube enormous amounts of material, enormous amounts of material.

Failure in Science: Mars Climate Orbiter (1999)



- Failure to do a proper conversion of units causes $330 million mission to vaporize in martian atmosphere
- *Consequence*: Public faith in scientific endeavor shaken

[8] James Oberg. "Why the Mars probe went off course". In: *IEEE Spectrum* 36.12 (1999), pp. 34–39.

left-pad incident[10]

- Node (`nodejs.org`)[9] is an open source Javascript (JS) runtime for creating command line tools and network applications
- The company npm, Inc. maintains a large software registry at `npmjs.com`
- In 2016, as a result of a dispute over naming, a developer unpublished a package called *left-pad* from the `npm` registry
- Many popular projects had *left-pad* as a dependency and its removal made the package management tool effectively unusable for many developers

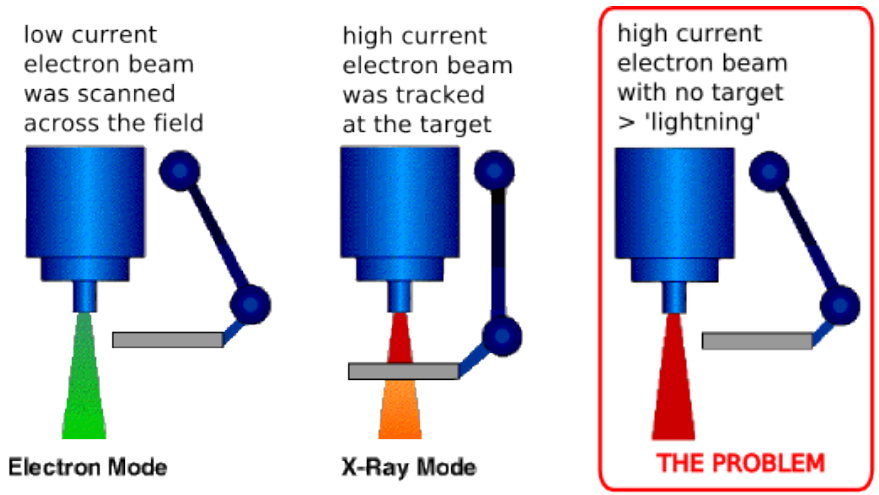
[9] Stefan Tilkov and Steve Vinoski. "Node.js: Using JavaScript to build high-performance network programs". In: *IEEE Internet Computing* 14.6 (2010), pp. 80–83.

[10] *The npm blog. kik, left-pad, and npm.*

Failure in the Entertainment Industry

- Many examples of projects with troubled development histories
- *Aliens: Colonial Marines*, *Assassin's Creed Unity*, *Duke Nukem Forever*, *The Last Guardian*, *Steam Launch*, ...

Therac-25[12]



tray including the target, a flattening filter, the collimator jaws and an ion chamber was moved OUT for "electron" mode, and IN for "photon" mode.

- Therac-25 Computer Controlled *Dual-Mode* Radiation Therapy Machine
- The image[11] above shows a schematic of a Therac-25 facility
- Software written by one engineer in PDP-11 assembly
- Malfunctions caused 6 people to receive massive overdoses of radiation, some resulting in death

[11] `http://web.mit.edu/6.033/2017/wwwdocs/assignments/rec-therac25.html`

[12] Nancy G Leveson and Clark S Turner. "An investigation of the Therac-25 accidents". In: *Computer* 26.7 (1993), pp. 18–41.

Hardware Safety Mechanisms

- An *interlock* is a type of safety mechanism that makes two states of a system mutually exclusive
- Examples include
  - Two-hand Control (pictured)[13]
  - Trapped Key Systems
- Previous Therac machines included both hardware and software safety mechanisms

[13] image credit: `osha.gov`

What is Software Engineering?

NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

Software Engineering (Sommerville)

The engineering discipline aimed at all aspects of producing software. Ideally, software should be high-quality and be produced cost-effectively.

Software Engineering (IEEE Std 610.12-1990)

The application of a systematic , disciplined, quantifiable approach to the development, operation and maintenance of software; that is, the application of engineering to software

- Important difference compared to other forms of engineering: *aeronautical*, *chemical*, *mechanical*, etc.
- Other disciplines generally talk about tangible systems

Thought-stuff

Fred Brooks, *The Mythical Man-Month*

The programmer, like the poet, works only slightly removed from pure thought-stuff.

- This quote comes from one of the most influential books on software engineering[14]
- The point is to emphasize that software does not suffer the same constraints as other forms of engineering...this is both a blessing and a curse
- Can be useful to use *metaphor*

[14] Frederick P Brooks Jr. *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition, 2/E*. Pearson Education India, 1995.

Metaphor of Construction[15]

- Sometimes people say that they *write* code...but do we write code like we write an essay
- Software engineering is often compared to building or construction
- What questions would you ask if you were leading a large-scale construction project?

[15] pictured: Glen Canyon Dam

IEEE Definition of a process[16]

The IEEE defines a process as "a sequence of steps performed for a given purpose" [IEEE-STD-610]

- A *process* is how you do something
- Everyone who codes follows some kind of *process* (even if they don't realize it)
- Our "default process" usually looks like *trial-and-error*:
  1. Google the problem
  2. Write some code, compile it, and then run it
  3. Try to debug if you encounter an error

[16] Mark Paulk et al. *Capability Maturity Model for Software (Version 1.1)*. Tech. rep. CMU/SEI-93-TR-024. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993.

Software Lifecycle Processes

- The "default process" is good when you are working alone but is not suitable for teams or for code reuse
- Every software project requires: *planning*, *designing*, *coding* ...
- Different processes organize these *activities* in different ways, this is the primary distinction between *waterfall* and *agile* processes
- Draw from the following software processes in this course
  - **Waterfall**
  - **Scrum**
  - **Extreme Programming**
  - **Lean**
  - Unified Process
  - Kanban
  - Crystal
  - ...

Practice, Principle, and Values

NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

Quote from XP Explained[17]

Good relationships lead to good business. Productivity and confidence are related to our human relationships in the workplace as well as to our coding or other work activities. You need both technique and good relationships to be successful.

- Software Engineering is largely about working in *teams*
- Any particular SE methodology will have *practices* that foster good teamwork
  - Example from XP: *pair programming*, the idea that all production code should be produced by two people sitting at one computer

[17] Kent Beck. *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

Organization

NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

Drucker, quoted by Lanham[18]

Management is about human beings. Its task is to make people capable of joint performance, to make their strengths effective and their weaknesses irrelevant. This is what organization is all about, and it is the reason that management is the critical determining factor.

- Software engineering is as much about human issues as it is about programming

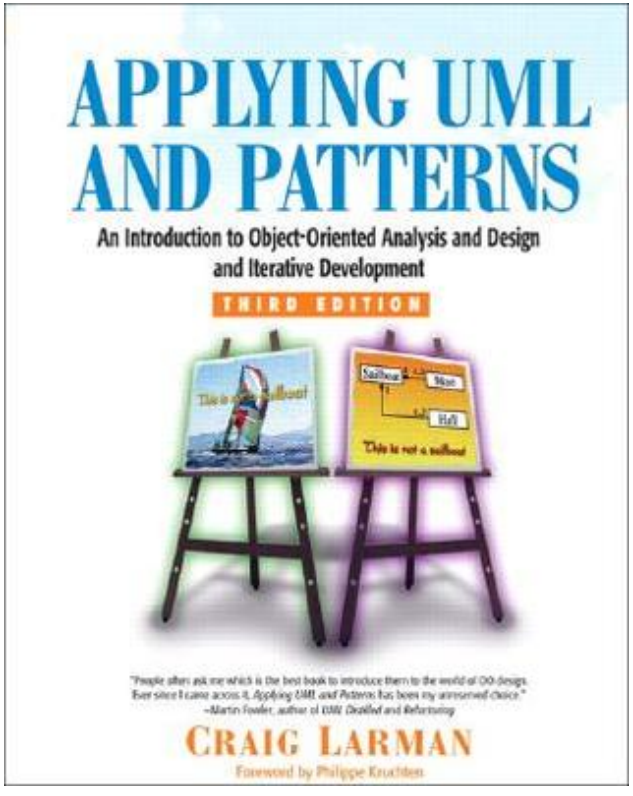
[18] Richard A Lanham. *The electronic word: Democracy, technology, and the arts*. University of Chicago Press, 2010.

Outline of the Course

- **General Learning Objective**: cultivate a skillful approach to software development
- Object-Oriented Programming and Design
- Software Processes (including Agile development)
- Requirements Engineering
- Tooling and Testing
- Modern Web-Development (Java EE)

Textbook

- The textbook for the course will be "Applying UML and Patterns" by Craig Larman
- In addition to the textbook, we have a detailed set of lecture notes, and readings available through the library resources



Materials

- We have compiled a large bibliography from scholarly and trade publications
- The course is Java-based (for review of Java I would recommend going through the OCA prep material[19])
  - The Eclipse IDE is available at URL `http://www.eclipse.org/ide/`
  - Maven is an apache project. Downloads and installation instructions can be found at `https://maven.apache.org/`
  - Git is available from `https://git-scm.com/` and Github, for hosting remote repositories is at `https://github.com/`

[19] *Java Tutorials: OCA I Preparation*.

- Complexity is one of the main challenges of software engineering and each style of programming (paradigm) has its own way of dealing with complexity
- In OO, we break up a complex system into distinct units (objects and classes) with their own well defined areas of responsibility
- A complex system can be implemented by *composing* many smaller units together
- In the OO section of the course we will cover *design patterns*, UML notation, and best practices (for example, the so called SOLID principles)

Grading Scale

NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

- Standard grading scale
  - A (95-100) A- (90-94)
  - B+ (85-89) B (80-84) B- (75-79)
  - C+ (70-74) C (65-69) C- (60-64)
  - D+ (55-59) D (50-54)
  - F (0-49)
- I will have the exact percentage breakdown in the syllabus: will include written exams and quizzes, homework assignments, and attendance
  - A significant portion of the grade will be based on a team semester project

Homework for this Week

- Review the Java Tutorials OCA I Preparation materials

  `https://docs.oracle.com/javase/tutorial/extra/certification/javase-8-programmer1.html`
- Download the required software: Java JDK, Eclipse, maven, and git
- Start thinking about forming teams with your peers (required size of teams will be 5-6 people)