COMS W4156 Advanced Software Engineering (ASE)

September 9, 2021
First day of class!

Lecture Notes linked on Courseworks Calendar

These are “live” links, so possibly changing up to and even during class

Questions and Comments during Class

We will start with Piazza Live Q&A but may switch to something else

Agenda

1. Brief overview of what I mean by “Software Engineering”
2. Overview of this course
3. Explanation of Homework Zero

What is Software Engineering?

Software Engineering != Programming

- *Time and Change* affect the sustainability of software, how code adapts over the length of its life
- *Scale and Growth* affect the viability of software practices, how an organization adapts as it grows and evolves

Time and Change

Google search was a graduate student project called "Backrub" in 1996, written in Java and Python [soon rewritten mostly in C/C++, with crawlers in Python]

In 1996, Java version 1.0, Python version 1.4

Today, Java version 16.x (many people still using 8 or 11), Python version 3.9.x (someone somewhere is probably still using 2.7 with "end of life" 2019/2020)

The original Backrub could not possibly run today

It is unlikely that any other 1996 or even 2006 software written in Java or Python (or most other languages) could run "as-is" today. Note this has nothing to do with new features or any other internal changes, the problem is *external* changes

Software Sustainability

Different from “sustainable computing” (aka “green computing”), although a mandate for green computing would count as an external change

How painful is it to modify the software to fit external change?

Could be change in language, compiler, libraries, operating system, hardware

Could be changes in regulations (e.g., GDPR and other privacy protections)

Could be transient changes due to world circumstances (e.g., sudden dramatic increase in unemployment insurance claims)

Success depends on organizational culture, process and tools

Exceptions

Some ancient programs survive

Scale and Growth

Backrub was written by Larry Page, Sergey Brin and a couple other Stanford graduate students

Their codebase was probably around 10k lines of code

At least 25k software engineers now work for Google

Current Google codebase contains billions of lines of code - but its still in a single monolithic codebase

Elaborate processes, tools and technology enable scaling

Human Scalability Reality Check

For course programming assignments where two people work together, it might be feasible to email new versions of the code back and forth

This usually fails by the time the team grows to four people - no one knows what the current version is or who has it

For course projects where four or five people work together using a shared github (or similar) version control repository, it might be reasonable to turn on notifications to automatically send email to everyone in the team when anyone pushes modified code

Imagine every software engineer at Google getting email whenever any other software engineer at Google pushes some code (about 16k commits per day)

How to Learn Sustainable and Scalable SE

Ideally, the entire class would work on finding and fixing bugs in 20 year old software consisting of a billion files, with new features constantly being added by tens of thousands of other developers, with four billion active users … but that's not going to happen

What will happen is:

- a short individual mini-project, where everyone modifies and tests the same small GUI web application*, which was written by someone no longer available*
- a larger team project (4-5 students per team), where teams propose, develop and test their own software service through two iterations - *services have APIs, not GUIs, no GUI projects allowed*

Visit Courseworks Home Page and Sections

Visit “Homework Zero”, due September 15

Or six days after you officially start the course (for waitlisted students)