CSE 486/586: Distributed Systems

Introduction

Ethan Blanton

Department of Computer Science and Engineering
University at Buffalo

Welcome to CSE 486/586

My name is Ethan Blanton

Contacting me:

Email:	eblanton@buffalo.edu
Office:	Davis 303
Office Hours:	Mondays 10–11, Wednesdays 14:30–15:30

The syllabus is available on the course web page, at https://www.cse.buffalo.edu/~eblanton/course/cse586.

So are these slides!

Expectations

For this course, I expect that you:

- Will be respectful to me, TAs, classmates
- Attend every lecture
- Adhere strictly to the academic integrity policy
- Will seek assistance early if necessary
- Meet prereqs; among other things:
  - Have some experience with network programming
  - Ditto systems programming
  - Understand data structures and algorithm analysis
  - Know or can rapidly learn Java

Most of all, behave as adults and strive to maximize your and your classmates' learning experience in this course.

Academic Integrity

I take academic integrity very seriously.

Violators will

- fail this course, and
- be referred upward for further sanctions.

I and the TAs will watch for violations.

Automated tools will be used to identify shared code.

Online resources (*e.g.*, Stack Overflow) will be monitored for copying.

Academic Integrity (*continued*)

You may:

- Seek help from instructors
- Discuss concepts with classmates
- Use code from Android Developers with clear attribution
- Use anything from the text with clear attribution

You MAY NOT:

- Share code with classmates
- Use code from *anywhere else*
- Discuss implementation with classmates

Academic Integrity — Good Practices

To ensure a pleasant semester, please:

- Be careful with permissions on code on GitHub, Bitbucket, shared UB filesystems, *etc.*
- Don't even look at each others' code!
- Cite everything
- Review the department and University policies

If in doubt, ask!

More on That Pleasant Semester

I intend for this course to be fun and rewarding.

You'll get out of it what you put in; no more, no less.

I do not take well to grade negotiation.

If you want a better grade, do better work.

If you're willing to put in the time, I'm willing to help.

Other Policies

Entire submissions or exams will be re-graded only for grading errors.

No incompletes will be given.

No makeup exams will be given.

No grades will be changed for any reason other than grading error.

Distributed Systems

From the text:

> *We define a distributed system as one in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages.*

This is obviously more interesting if they work together.

Distributed Systems

Some more adjectives we will use for these systems:

- autonomous
- programmable
- asynchronous
- failure-prone

Furthermore, we will assume that the network is unreliable.

Why Distributed Systems are Hard I

Scale: many machines, huge data

- Google Percolator is designed for "1000s of machines" [5]
- Pinterest databases have hundreds of shards, store petabytes of data [9]
- Facebook page loads touch more than 1k servers [3]
- Facebook's Hive data warehouse was 300 PB in 2014 [8]

Why Distributed Systems are Hard II

Failures: systems are unrealiable

- Backblaze reports drive failure rate of 2%/year [4]
- Google reports uncorrectable errors in 20% or more of SSDs in their first year of service [6]
- At Google, “In a single cluster in a typical year, thousands of machines fail and thousands of hard disks break”. [1]

Why Distributed Systems are Hard III

Concurrency: coordination is complicated and necessary

- 400 hours of YouTube video is uploaded every minute [7]
- Facebook users like over 4 million posts per minute [2]
- Search engines like Google and Bing perform continuous updates on an enormous index — while users are searching

Course Materials and Activities I

Materials to learn from:

- Lectures
- The text: *Distributed Systems: Concepts and Design (Fifth Edition)* by Coulouris, Dollimore, Kindberg, and Blair.
- Assigned (required) readings
- Suggested (optional) readings

Course Materials and Activities II

Activities to learn from:

- Projects
- Homeworks (assigned but not graded)
- Exams

Programming Projects

A significant portion of your course grade will be projects.

- These are individual projects.
- Projects will be Android Java applications.
- They will be evaluated in an emulator.

GitHub Classroom

We will use GitHub Classroom

- for assignment distribution
- for providing assistance

You must have (or create!) a GitHub account.

You are expected to use git and GitHub for development.

E.g., TAs won't look at code unless it's on GitHub!

Info:

- Android Studio has git support: tutorial, StackOverflow GitHub imports
- Git help: Git book, tutorial, Google

Project Assistance

Your TAs will be your primary source of help for projects.

To get the most out of your TAs, do:

- try the obvious things first,
- create minimal examples to show problems, and
- consult the documentation.

To avoid wasting TA time and failing to get help, don't:

- ask for help before you've tried to understand the problem, or
- start at the last minute.

Project Submission

We will submit using Autograder.

Submission rules:

- Submitted w/in 24 hours of the deadline: -20%
  - Doesn't count Saturday or Sunday
  - Doesn't count University holidays
- Projects submitted after 24 hours will not be accepted

Example 1: Project is due Friday at 11:59 PM, turned in Monday at 3 PM — 20% penalty.

Example 2: Project is due Monday at 11:59 PM, turned in Wednesday at 12:15 AM — not accepted.

Project Description

Your first project will be a “starter project” to:

- Familiarize you with Android development
- Give you a self-evaluation for course preparedness

The next three (four?) projects will:

- Build upon one another
- Work together to implement a key-value storage system in the style of Amazon Dynamo
- Cement and evaluate your understanding of key distributed systems concepts

Project 1

This project is a readiness test for CSE 486/586.

- If you cannot complete it on your own in one week, you will have trouble in this course
- It will be due one week from today @23:59

It is a simple messenger app for Android.

- Figure out GitHub Classroom
- Set up Android Studio
- Understand code handout
- Use sockets
- Use some Android APIs and create event handlers

Today's Assignments

As soon as possible:

- *Join our GitHub Classroom*. The first programming assignment requires this!
- Read the syllabus.

By **Friday, February 2 at 23:59:00**:

- *Complete the Academic Integrity Quiz* at `https://www.cse.buffalo.edu/~eblanton/misc/academic_integrity/` and turn it in on Autograder.

By **Monday, February 5 at 11:59:00**:

- *Complete Programming Assignment 1* and turn it in on Autograder.

Next Time ...

- A refresher on Internet architecture and network communication

References I

Optional Readings

[1] Betsy Beyer et al., eds. *Site Reliability Engineering: How Google Runs Production Systems*. O'Reilly Media Inc., 2017.

[2] George Carey-Simos. *How Much Data is Generated Every Minute on Social Media?*. http://wersm.com/how-much-data-is-generated-every-minute-on-social-media/. Aug. 2015.

[3] Jillian D'Onfro. "Peek inside Facebook's massive data centers that store all your photos, 'likes,' and chats". In: *Business Insider* (Sept. 2015).

[4] Andy Klein. *Backblaze Hard Drive Stats for 2016*. Backblaze blog entry for January 31, 2017.

[5] Daniel Peng and Frank Dabek. "Large-scale Incremental Processing Using Distributed Transactions and Notifications". In: *Operating Systems Design and Implementation*. USENIX. 2010.

References II

[6] Bianca Schroeder, Raghav Lagisetty, and Arif Merchant. "Flash Reliability in Production: The Expected and Unexpected". In: *File and Storage Technologies*. USENIX. Feb. 2016.

[7] Kevin Tran. *Viewers find Objectionable Content on YouTube Kids*. http://www.businessinsider.com/viewers-find-objectionable-content-on-youtube-kids-2017-11.

[8] Janet Weiner and Nathan Bronson. *Facebook's Top Open Data Problems*. https://research.fb.com/facebook-s-top-open-data-problems/. Facebook, 2014.

[9] Yongsheng Wu. *Evolution of Storage and Serving at Pinterest*. Data @Scale. 2017.