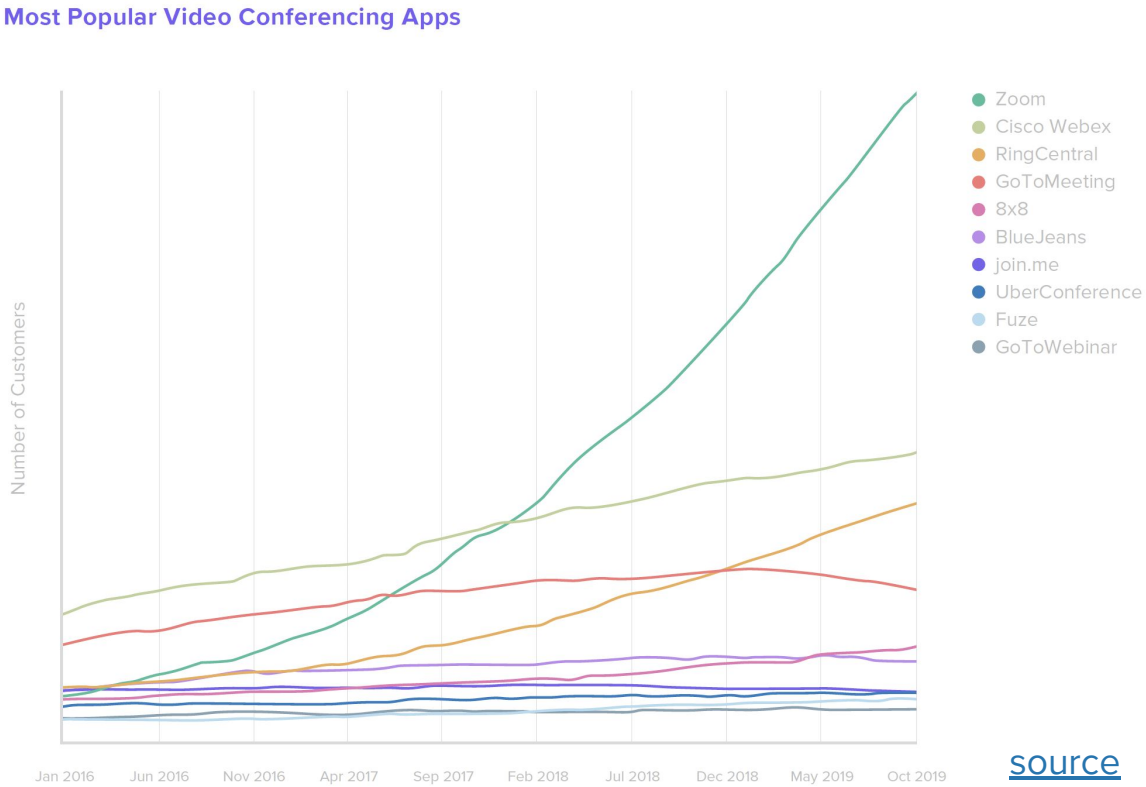
CS188
Scalable Internet Services

John Rothfels, 10/1/20

Motivation



source

Zoom Outages: 5 Reasons Why Zoom Is Experiencing Growing Pains

From a couple of widespread Zoom outages, to security and data privacy concerns, here are five reasons that could explain Zoom's growing pains over the last five months.

By **Gina Narcisi** August 25, 2020, 03:00 PM E

Unprecedented Growth

As the global pandemic forced many employees to work from home and students to distance-learn as schools and universities closed, Zoom saw a "massive increase in demand" for its services in a very short period of time. That uptick in demand required rapid changes to be made by Zoom's developers in order to have the scale in place necessary to accommodate millions of new users right away. But scaling up quickly can result in networking or security issues.

According to Downdetector.com, there were more than 15,000 reports of problems with Zoom video as of 9:48 a.m. ET on Monday. The site classified 76 percent of those problems as log-in issues. All other Zoom capabilities, including Zoom Phone, Chat, its conference room connector, cloud recording, meeting telephony services, and the Zoom developer platform were reported as operational.

Zoom also experienced a widescale outage in early April with users on the East Coast of the United States and parts of Europe reporting error messages upon attempting to log in to the Zoom web client. To a lesser extent, the outage was felt in parts of California, Florida, and the Midwest, as well as Malaysia, according to DownDetector.com.

source

Motivation

You’ve built something that everyone is excited about.

It gets popular.

Its popularity doubles.

Then it doubles again.

Then it doubles again.

…

Motivation

You’ve built something that everyone is excited about.

It gets popular.

Its popularity doubles.

Then it doubles again.

Then it ~~doubles~~ crashes again and again and again… 🔥 🔥 🔥

Then you wish you’d taken CS188! 🙏

Motivation

Let’s say…

… I want to find a home to live in…

… I’m lost in a foreign city…

… I want to go on a date…

… the world is shut down due to COVID-19…

… what do I do?

Motivation

Every day, billions of people use the same suite of technologies to solve these problems: **internet services**.

As these services get increasingly popular, they need to continue to function.

⚠️ Scaling relatively simple web applications can be very complex. ⚠️

Internet services

For the purposes of this class, by **internet services** we mean HTTP services.

ℹ️ There are other internet protocols besides HTTP. ℹ️

Internet services

For the purposes of this class, by **internet services** we mean HTTP services.

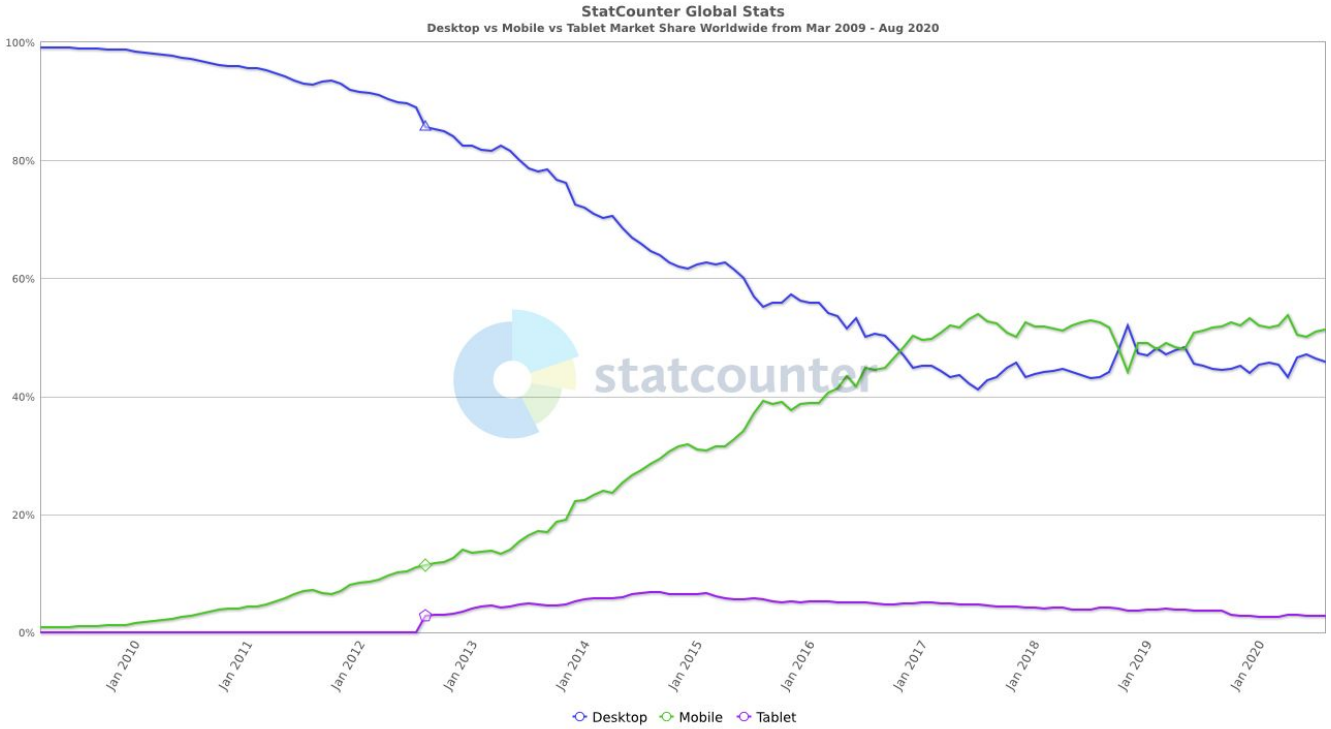
Do we mean HTML?

HTTP encompasses more than just HTML. HTTP is the foundation for most APIs on the internet today. REST, SOAP, and GraphQL are conventions on top of HTTP for machine to machine communication. HTTP is a go-to solution for data exchange.

Do we mean web browsers / mobile devices?

Yes! Both communicate with internet services over HTTP.

Internet services



source

Scalable internet services

An internet service is **scalable** if increasing demands can be effectively met with increasing capacity.

By **demands**, we usually mean **traffic**.

ℹ️ Other demands apply as well, such as size of data or compute requirements. ℹ️

Scalable internet services

An internet service can **effectively meet** demands if:

- Service stays available.
- Response time doesn’t excessively degrade.

Scalable internet services

An example internet service:

- Run everything on one server.
- When traffic gets too high, we buy a bigger server.

Is this scalable?

Course overview

Scaling internet services is the core of the course. We'll look at a variety of topics along the way.

Learn by doing! You will build, deploy, (break), and scale your own internet service.

This course won't teach you how to get worldwide attention. It *will* teach you how to build a system that can respond to worldwide attention.

Course overview

⚠️ This class is a mile wide and an inch deep! ⚠️

This is **not** a deep-dive in relational databases, networking, distributed systems, network security, or cloud computing...but we'll touch on each of these.

For deep-dives, see:

- Networking: 118, 117, 211, 218
- Distributed Systems: 133
- Databases: 143
- Network Security: 136

Course structure

Lecturer: John Rothfels (rothfels@cs.ucla.edu)

- Director of Engineering, Dynasty/AppFolio
- Teaching @ Stanford, 2009 - 2012

TAs:

- Salekh Parkhati (ssaa@ucla.edu)
- Daniel Achee (dpachee@cs.ucla.edu)

Course structure

Lectures:

- Tue/Thu @ 8am.
- Covers concepts required for labs/projects and more. Live demos!

Labs:

- Fri @ 12pm, 2pm.
- Focused on the course project.
- Demo progress to instructor/TAs each week.

Course structure

Website: https://cs188.cloudcity.computer

Piazza (discussion forum): https://piazza.com/class/kfpm567u1e24eb

- Email notifications are a good idea.
- Helping others is very strongly encouraged. 🙏❤️

Your project: https://**yourproject**.cloudcity.computer

Course structure

There are no course textbooks. We'll use free online content only.

Readings will be assigned for some lectures. Check the Lectures page for links.

Other useful readings / resources are linked from the course homepage.

Course structure

Your grade will be based on 3 primary factors:

- Your final project presentation.
- Your final project paper.
- Your teammates' evaluations of your contributions.

Also considered, to a lesser degree:

- Weekly lab demo progress.
- Assistance to other students on Piazza.

Course project

This course is **project intensive**. Learn how to build a scalable service by building one! You will...

- Work in teams of 4.
- Develop an interesting internet service.
- Deploy it on AWS.
- Measure its performance/scalability.
- Apply the techniques presented in class to improve it.
- Document these improvements and present them.
- Share with your families and friends!

Course project

This course demands a lot of work, but it will be **very rewarding**. Things you will learn in the next 10 weeks:

- A programming language (TypeScript / JavaScript).
- An application development framework (React / GraphQL).
- How to deploy and run code on Amazon Web Services (`terraform`).
- How to load test an application.
- How to debug performance problems in production.
- Agile software development.

Course project

This course & project has an **industrial focus**. We will use industrial software development techniques.

All project code will be open-source. Learn from your classmates!

Toolchain we'll be using (100% open-source software):

- TypeScript, Node.js, MySQL, Redis, React, REST/GraphQL, WebSockets

Course project

Why Node.js?

- **Pros**:
  - Single-threaded
  - Fast
  - Simple to get started
  - JavaScript (?)
- **Cons**:
  - Single-threaded
  - Callback hell
  - Poor stack traces
  - JavaScript (?)

Course project

Why TypeScript?

- **Pros**:
  - Static analysis/tooling
  - Works on browser & server
  - JavaScript interop
  - Build toolchain
- **Cons**:
  - Not JavaScript (more verbose)
  - Built by Microsoft (?)
  - Not <your favorite server language>

Course project

Why React?

- **Pros**:
  - Industry-wide adoption: it's more popular than orange juice 🍊
  - Advanced tooling/features
  - Simple design model
  - Client-side rendering
- **Cons**:
  - Large footprint (bundle size)
  - Built by Facebook (?)
  - Less familiar than traditional server-side rendering / HTML
  - Not <your favorite frontend framework>

Course project

Why Node.js, TypeScript, React?

In order to learn advanced scaling topics, you need to get a project from zero to working quickly.

This class is built around an application framework **designed specifically to let you explore these concepts quickly**.

⚠️ The technologies we'll use are industry-standard, but others exist. ⚠️

Course project

What's in it for me?

If you're going into academia, knowledge of industrial software engineering is valuable as it sets the context for important problems. 🚀

If you're headed for industry, there's never been a better time to understand this stack. 🚀

Course project

What's in it for me?

- You can build something that interests you! 😍
- Show it off! We'll have a public exhibition at the end of the quarter.
- Keep building your project after the quarter is over, if you like.
- You can work with an interesting and large dataset.
- You get to practice working with other people and using industrial software engineering tools and techniques.

Course outline

The **lectures** will cover material you need to build a successful project, and additional material that won't be directly applied.

Necessary content for successful projects is front-loaded so you can apply these concepts early.

The lectures in the second half of the quarter will be additional material and guest lectures.

Course outline

The **lectures** in the first 5 weeks will cover the core of the course.

- Intro to the basics: HTTP, HTML, JavaScript, CSS
- HTTP / application server architectures
- API design: REST, GraphQL
- High availability and load balancing
- Horizontal and vertical scaling
- Client-side and server-side caching
- Using and scaling relational databases via sharding, SOA, and read replicas
- Observability and monitoring
- Load testing

Course outline

The **lectures** in the subsequent 5 weeks will cover additional topics and include guest lectures.

- Scaling with non-relational datastores
- Serverless computing
- HTTP/2.0
- Industrial software engineering: agile, TDD, CI/CD, pairing
- Thicker clients: Asm.js, Emscripten, WebAssembly
- Intelligent web systems using machine learning
- CDNs, edge networks

For next time

Assigned reading for next lecture on course website.

Start thinking about project ideas and groups. You will need a group by the start of next week's lab. Use Piazza to look for a project group.

Lab tomorrow:

- get project starter code
- setup dev environment
- learn TypeScript/React