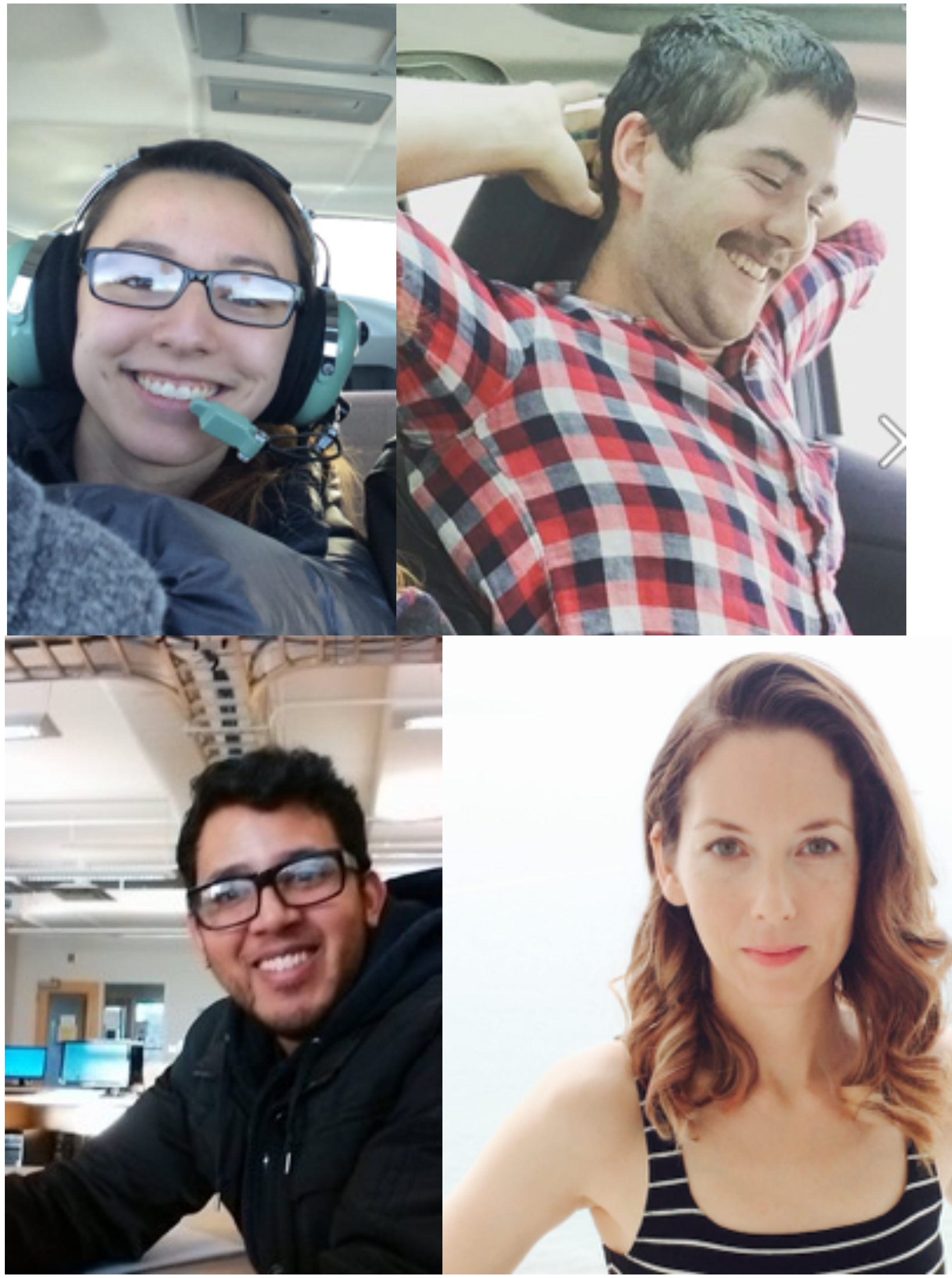
Distributed Systems
CPSC 416
Winter 2017

Course: January 4 - April 5, 2016

Jan 4, 2016 Lecture (first class!)

Course staff

- Ivan Beschastnikh, instructor
- TAs
  - Amanda Carbonari (1/2)
  - Stewart Grant
  - Rohin Patel (1/2)
  - Jodi Spacek



Logistics

- Last year the course had ~77 people
- This year we are at 117
    - Added a TA
    - Dropped project
    - Added (many) assignments

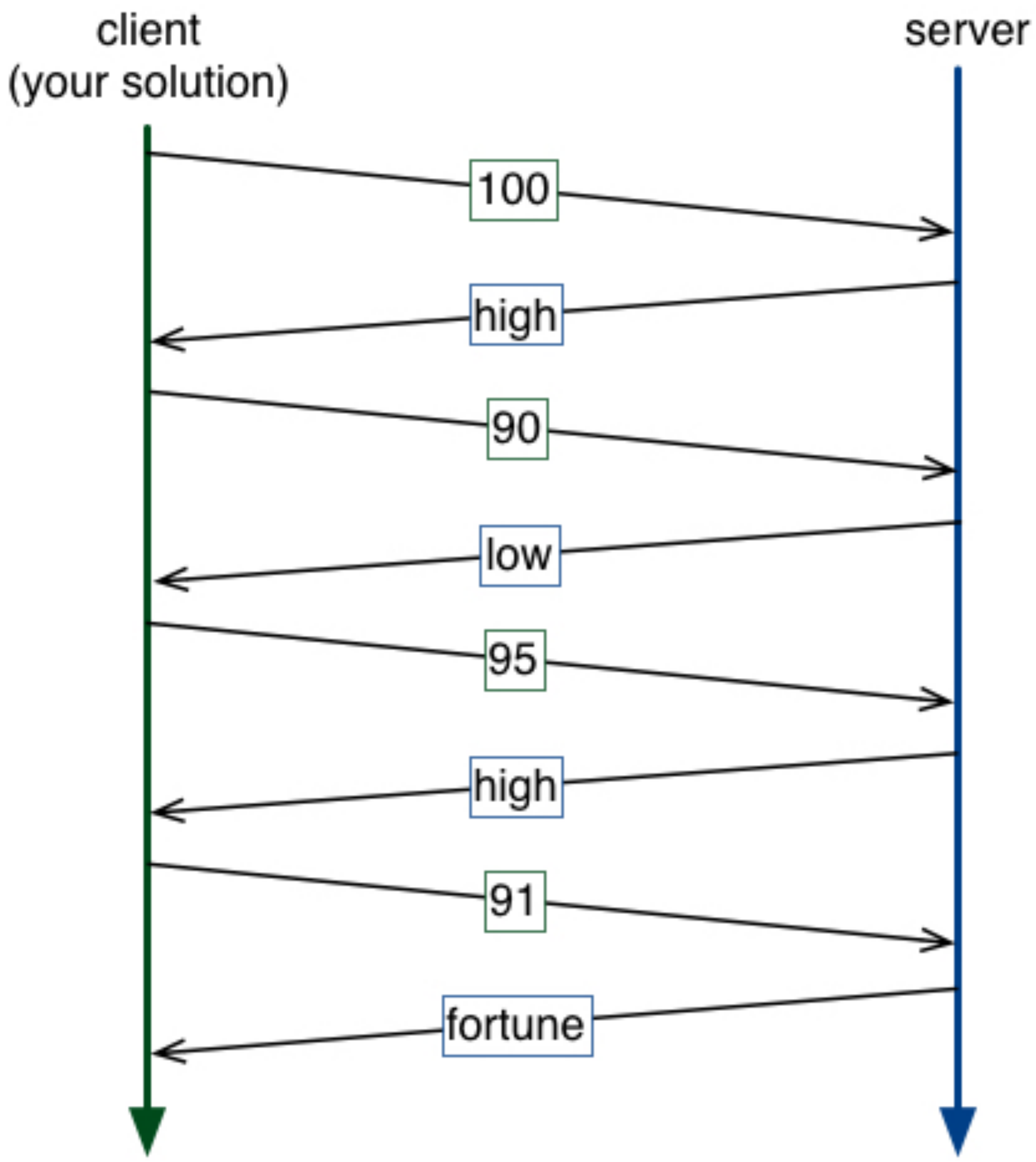
Logistics

- Everything on the website, updated continuously:
  http://www.cs.ubc.ca/~bestchai/teaching/cs416_2015w2/index.html
- Use Piazza for all course-related communication

Course overview via the website

- Learning goals
- Go programming language (start learning!)
- Schedule (a work in progress)
  - Assignment 1 due Jan 13 (next Wed)
- Exam (just a final)
- Advice for doing well
  - learn Go (a must to pass the course)
  - don’t hack, engineer
  - choose team, wisely
  - reach out on Pizza/email for help.
- Collaboration guidelines

Assignment 1: Goldilocks fortune (due week from Friday)



Assignments note

- Last year’s 416 TA rant:

TEST YOUR CODE ON THE UGRAD MACHINES!!!!!!!!!!!!!!!!!!!

YOU WILL GET ZERO IF IT DOESN'T RUN OR COMPILE. WE HAVE NO SYMPATHY FOR THESE TYPES OF ERRORS.

… you’ve been warned

Distributed system examples

- YouTube
  - Videos are **replicated** (multiple machines host the same video)
  - **Scalable** wrt. client requests for videos (internally **elastic** — can throw more machines at the service to have it scale out further)

Distributed system examples

- DropBox (or google drive)
  - **Replicated** content across personal devices
    - Supports **disconnected operation** (can work while disconnected, and synchronize when re-connected)
    - Maintaining data **consistent** across devices
  - Supports sharing; **access control** policies (security!)

Distributed system examples

- NASDAQ
  - **Transactions** (e.g., ACID semantics from databases). Many DBMS concepts apply to distributed systems!
  - Strong **consistency** and **security** guarantees (otherwise people would not trust it with money)

Some D.S. challenges

- Synchronizing multiple machines (protocol complexity)
- Performance (how do you define/measure it?)
- Maintaining consistency: strong models (linearizable) to weak models (eventual) of consistency
- Failures: machine failures (range: failure stop to byzantine); network failures (just a few: disconnections/loss/corruption/delay/partitioning)
- Security (how to prevent malicious control of a single host in a system escalating into control of the entire system?)