Distributed Software Development: Introduction

Chris Brooks

Department of Computer Science
University of San Francisco

Class structure

- Combination of lectures and labs
  - Labs (many Tuesdays): hands-on exposure to a piece of Web technology.
  - Lectures (Thursdays): problems, principles and algorithms related to large-scale distributed systems.

Class struc

- Work:
  - 8 labs - 1 week turnaround.
  - Midterm and final
  - Two projects:
    - P2P client
    - Extension of your choosing.

Course Policy

- Class participation is very important.
  - Attendance is required
  - Active participation is encouraged (and rewarded!)

Course Policy

- Texts:
  - Readings from a number of texts are available online through the USF library
  - I will also assign readings from O'Reilly's technical collection.
  - This is available for free to all USF students at proquest.safari.com
  - I will expect you to do the reading before class.

On Reac

- "There's so much reading! Do we really have to read all of this?"
- "Can't we just use the slides?"

On Reading

- "There's so much reading! Do we really have to read all of this?"
  - Yes.
- "Can't we just use the slides?"
  - No.

On Reading

- One thing I want you to learn in this class is how to synthesize large amounts of printed information.
- Wrong: Front-to-back, sequential.
- Right: Skim first. Identify important sections, or material you are weak on. Return to that.

Course Po

- Languages:
- You may use any language that "makes sense" for the lab/project.
  - I will present examples in Python and/or Java.
  - If you want to use something esoteric (not Python, Java, C/C++, Ruby, Perl, or C#) please talk to me first.
  - I recommend choosing a language that has good support for XML parsing.

On Plagiarism

- You are all expected to do your own work. No exceptions.
- There may be cases in which it is acceptable to use a third-party library (with appropriate credit) to assist on an assignment. Please ask first.
- I will check your assignments against each other, databases of code repositories, and google.
- Plagiarism is simply not worth the risk.

Assignments

- You will have 8 labs, plus two larger programming assignments.
- Labs will provide you with some components that can potentially be used in larger projects.
  - Good SE practices are important here.
- All labs **must** be available 24/7 on a publically available server.
- Pace is fairly quick; it's very important to start early and keep up.

Course Po

- Late assignments:
  - Labs may be turned in up to 24 hours late for a maximum of 75% credit
  - Projects may not be turned in late. You will be expected to demo your work the day it is due.
- In general, getting in the habit of turning things in late is a very bad idea.

Aphorisms

- "Success is 1% inspiration and 99% perspiration" - Thomas Edison
- "90% of life is showing up." - Woody Allen
- "Teachers open the door. Students must walk through on their own." Chinese proverb.
- "Just keep swimming." Finding Nemo

What is a distributed system?

- (Coulouris) "A distributed system is one in which hardware or software components communicate or coordinate their actions only by passing messages."
- This covers everything from a parallel computer to the Internet.

What is a distributed syst

- So how is this class different from Pacheco's or Benson's?
- Different set of challenges:
  - Heterogeneity
  - Openness
  - Scalability
  - Failure models
  - Degree of parallelism

Tightly-coupled systems

- Parallel computing typically works with the following sorts of systems:
  - Shared clock
  - Low latency/fast communication
  - Single owner
  - Homogenous systems
  - Tightly coupled
- Prof. Pacheco's class addresses systems with these characteristics.

Middle ground

- Prof. Benson's class relaxes these assumptions somewhat.
  - Systems may not be homogenous
  - Communication still fast/low latency
  - Limited autonomy
  - Less tightly coupled.

This c

- This class will look at systems on the other end of the spectrum
  - Widely distributed - high latency communication
  - Autonomous, heterogenous components
  - No shared clock
  - Very large scale
  - Discovery may be a problem
- The Web is a classic example of this sort of system (but not the only one)