COMP 3010 - Distributed Computing

Calendar Description: An introduction to the development of client server and peer-to-peer systems through web applications, distributed programming models, and distributed algorithms.
Prerequisite: COMP 2150.

Outline

1. Introduction and Motivation (2 weeks)
   A discussion on the need for distributed systems, fundamental principles and standard issues. Topics include the basics of communications, resource naming and location, failure modes and scalability. We also introduce the idea of scripting languages for implementing distributed applications along with the encoding of data for transmission.
2. Web-based Computing (4 weeks)
   A discussion of the HTTP protocol and server-side processing. The basic messaging scheme is discussed along with the transmission of form data via GET and POST methods. Server-side processing includes discussions on CGI implementation, server side includes, server pages (such as JSP), cookies and session data management. Database are not discussed since the emphasis is on the processing needed; students should be able to apply the material with that of 3380 to implement a "proper" web application.
3. Distributed Programming (3 weeks)
   A look at the implementation of distributed applications. We start with a look at the standard client/server architectures and how these design decisions affect implementation. Threads are not discussed even though they are required for proper server implementation; students should be able to apply the material with that of 3430 to implement a complete server. We look at different programming models including sockets, RPC and RMI. These require a discussion on IPC models.
4. Distributed Algorithms (4 weeks)
   A look at classic algorithms and how they reflect on current distributed computing problems (a lot of the algorithms link directly to peer-to-peer computing). Algorithms include consensus/mutex, failure detection and recovery (including Byzantine generals, election and ring algorithms), distributed time (both Lamport's and Fidge's algorithms), and distributed deadlock.

Text: none