Ser321 Principles of Distributed Software Systems

1. Course Coordinator
   Tim Lindquist

2. Catalog Description
   Principles underlying design and implementation of distributed software components; sockets, protocols, threads, XML, serialization, reflection, security, and events. Prerequisite: Ser222 and MAT243.

3. Credits and Contact Hours
   3 credits, 3 contact hours

4. Prerequisites or co-requisites by topic areas.
   Basic Web page construction in HTML.
   Data structure design and implementation techniques;

5. Course Overview:
   The course explores the concepts and techniques for designing, constructing and deploying distributed and Web-based applications. It examines the basis techniques of secure multithreaded, and distributed programming, and covers the most common distributed computing paradigms along with the frameworks that support the paradigms. Students will develop distributed applications using languages such as Java, C#, C++, and Objective-C. The course covers peer-to-peer, group communications, client-server, and distributed object based paradigms.

6. Course Resources

7. Course Student Outcomes (CO) and Mapping to Program Student Outcomes
   CO-1. Develop distributed applications - that are multi-language applications; such as: Java, C# .NET, C++, and C/Objective-C (Program student outcome: Technical Competence).
      Sources: (key performance indicators) midterm and final exam questions; programming projects 2,3,4.

      Source: Project assignment 3, and 4. Individual threaded socket server and client.

   CO-3. Utilize externalization, XML and security facilities to create robust and secure distributed applications. (Student outcome: Technical Competence).
      Source: Project assignment 1, midterm exam questions.
CO-4. Understand the concepts of threads and concurrency for use in distributed applications. Have the ability to create threaded servers and clients using best practices. (Student outcome: Technical Competence).
   
   Source: Project assignment 3 and 4. Final exam questions.

CO-5. Utilize fundamental program development tools -- command-line interpreter, compiler, debugger, make-facilities to develop, deploy and test distributed applications consisting of multiple programs and modules. (Student outcome: Technical Competence).
   
   Source: Programming projects 0-4. Midterm exam questions.

8. Topics
   a. Prerequisite review and introduction to tools used in the course.
   
   b. Introduction to distributed and Web-based applications.
   
   c. Binary, custom and XML serialization.
   
   d. Concurrency, threading, monitors, and synchronization.
   
   e. Inter-process communications, primitives, connection and connectionless, data marshaling.
   
   f. Distribution, sockets, threaded servers, clients, and thread pooling.
   
   g. Securing distributed applications: secure communications and securing the runtime environment.
   
   h. Distribution paradigms such as: client-server, peer, service-oriented, cloud.
   
   i. Distributed object applications and web services. Distribution object management: stateless, singleton and session. Parameters, return types, proxies and dynamic calling and dynamic code loading.
   
   j. Exams: Midterm exam covering first half of the semester; final exam covering all material.

9. Assignments:
   a. Individual programming project to successively develop a distributed application with multi-language clients.
   
   b. Team-based distributed application security and packaging assignment.

10. Grading:
   The projects in the course involve software design and implementation, teaming and communication skills. Semester long individual project and team project contributes to 40% of the grade. Two exams (midterm and final) account for 30% each.

11. Course Offering Information
   1. Desired instructor expertise: background in team formation and dynamics; experience developing distributed and web applications on multiple platforms including windows mobile, MacOSX, iOS, Android and/or Unix utilizing multiple high-level programming languages.
   2. No articulation information since the course is upper-division.
   3. Offerings in Fall semester.
   4. Course is offered only on 15-week semester schedule due to extensive individual code design and development required for the distributed application.