VANDERBILT UNIVERSITY
DEPARTMENT OF BIOMEDICAL INFORMATICS
COURSE SYLLABUS
FALL

I. General Information

A. Course: BMIF 340. Clinical Information Systems and Databases

B. Department: Department of Biomedical Informatics

C. Pre-requisites: Basic competency in using word processing, e-mail, and presentation software. Coding ability in a procedural or object-oriented programming language.

D. Description: This course introduces the student to Clinical Informatics, and specifically clinical information systems and databases. Design and development of software artifacts is used to provide an in-depth understanding of the current state of the art in Clinical Informatics. The course includes concepts of: Hardware and software aspects of computation: Moore's law; computing theory, computational complexity; operating systems; client-server architectures.

Computer networks: evolution of networked computing; OSI stack; network protocols; TCP/IP; TCP sockets; IP addresses and domain names.

Distributed computing: Synchronization, concurrency, scalability. Inter-process communication; name services; distributed and centralized monitoring. Synchronous and asynchronous communication.

Clinical information systems: historical evolution; trends; major applications; architectural overview.

Health Information Systems architectures: vertical vs. horizontal applications; data repositories; integrated architectures; interface engines; data interchange protocols.

Clinical databases: purpose and functions; characteristics of health data; uses of health data; electronic patient records. Case study: the VUMC clinical database architecture.

E. Credit: 3 semester hours didactic.

F. Course Faculty: Dario Giuse, Dr. Ing. Associate Director, Informatics Center; Associate Professor of Biomedical Informatics.

Office: 410 Eskind Biomedical Library.

Phone: (615) 936-1435

e-mail: dario.giuse@vanderbilt.edu

Pager: (615) 835-1712

II. Course Objectives

By the end of the course, students will:

A. Know about the major network communication protocols, and have used network-based communication to implement simple distributed software systems.

B. Demonstrate understanding of distributed software systems, the client-server paradigm, and demonstrate the ability to evaluate the architecture of a software system in terms of robustness and scalability.

C. Demonstrate knowledge of the HL7 messaging protocol and be able to extract clinical data from HL7 messages.

III. Course Requirements

A. Learning Experiences:

Assigned reading: Students are expected to do the readings before each class and come to class prepared to discuss the content.
Class sessions: Students will raise questions about any points that were not clear in the reading. Classes will be highly interactive, and students will take turns leading discussions.
Course projects: Students can work on course projects individually or in groups, by previous agreement with the instructor. If more than an individual works on a project, all group participants must demonstrate complete knowledge of any portion of the project, and each participant assumes complete responsibility for the overall success of the project.

B. Learning Resources:

Textbooks (recommended):

- Wall L, Christiansen T, Schwartz RL. Programming Perl. 2nd Edition. O'Reilly & Associates, Inc. *This is the "bible" for Perl programming; this is the Perl book you should have if you only own one.*
- Flanagan D. JavaScript – The Definitive Guide. 4th Edition. O'Reilly & Associates, Inc. *Ditto for JavaScript programming.*
- Shortliffe, E.H. and Perreault, L.E. (eds.) Medical Informatics: Computer Applications in Health Care and Biomedicine. 2nd Edition. New York: Springer.
- Willis T, et al. Professional VB6 Web Programming. Wrox Press Ltd. *Contains a good introduction to networks and distributed systems, plus a wealth of information on the Microsoft software architecture.*
- Stein LD. Network Programming with Perl. (2000) Addison-Wesley. *The classic book for Unix-style distributed systems. Assumes good knowledge of the Unix programming interface.*

IV. Evaluation

Participation in class: 20% of course grade.
Project assignments: 70% of course grade.
Written reports: 10% of course grade.

V. Lectures (preliminary)

Introductions, course overview, logistics; discussion of learning materials; overview of course projects.

Fundamentals of computing theory. Recursive functions, λ-calculus, Goedel numbers, Turing machines. Undecidability and computational complexity.

The evolution of networking protocols.

Fundamentals of computer networks.

Fundamentals of computer networks.

Fundamentals of operating systems.

Fundamentals of operating systems.

Berkeley sockets.

Server schemes: single-shot, forking, pre-forking. Process creation, virtual memory, copy-on-write. Creating a child process: fork, exec, system.

The simplest socket-based server. Assignment: write a non-forking HTTP server.

Processing HTTP requests in Perl.

HTTP protocol details. The CGI protocol.

HTTP-based name server (using file system as the storage scheme). Synchronization in distributed systems.

Using Domain Name Services to provide reliable access to a distributed system. Discussion of single-point-of-failure and round-robin policy for load balancing. Interaction of connection-oriented sessions and round-robin name services.

The VUMC Clinical Information Systems architecture, circa 1995. Vertical ("stovepipe") clinical applications. Assignment: implement an HTTP-based name server, using the file system for storage.

The VUMC Clinical Information Systems architecture, 1995-1996. Provider order entry strategies and solutions.

The VUMC Clinical Information Systems architecture, 1996-1997

The VUMC Clinical Information Systems architecture, 1997-1998. Assignment: implement a replicated name service. Each student implements an HTTP-based name server that cooperates with the other students' to provide a robust service.

The VUMC Clinical Information Systems architecture, 1999-2001

The MARS clinical repository.

From MARS to StarChart. Assignment: store the training database patient data in a distributed service; develop a new service which allows query of the form "retrieve the most recent problem list for a patient".