

CS351 – Systems Programming

Last Updated - 03/01/02

Course Manager – Dr. Phil Dickens, Assistant Professor

3 credit hours; required for CS & CPE; 100 min. lecture & 100 min. lab each week

Current Catalog Description - Examines the components of sophisticated multi-layer software systems- including device drivers, systems software, applications interfaces, and user interfaces. Explores the design and development of interrupt-driven and event-driven software. Prerequisites: CS 331, CS 350. (2-2-3)

Textbook

- Charles Petzold, *Programming Windows*, 5th edition, Microsoft Press

References - other textbooks or materials

- none

Course Goals - Students should be able to:

- Design, code, test, and debug simple event-driven programs that respond to operating system and user generated events.
- Design, code, test, and debug applications with multiple objects communicating via message passing.
- Use a GUI toolkit to create a simple application that supports a graphical user interface.
- Explain the value of application programming interfaces (APIs) in software development.
- Design, code, test, and debug reasonably large software systems using native APIs.
- Design, code, test, and debug simple applications using threads.
- Design, code, test, and debug simple client-side applications using asynchronous socket programming techniques.

Prerequisites by Topic

- Data structures
- Computer organization
- Assembly language

Major Topics Covered in Course

1. Introduction to asynchronous programming	6 hours
2. Introduction to Windows programming	6 hours
3. Windows Graphical Device Interface	8 hours
4. Scroll Bar Messages and Manipulation	4 hours
5. Basic Concepts of "Virtual" Devices	4 hours
6. Handling Input Devices	4 hours
7. System Timers	2 hours
8. Developing Child Window Controls	4 hours
9. Basics of Menus, Dialog Boxes, and Resources	4 hours
10. Introduction to Threads	4 hours
11. Thread synchronization	5 hours
12. Network Programming	8 hours
Midterm Exam	1 hour
Final Exam	-
	60 hours

Laboratory projects (specify number of weeks on each)

- 1) Introduction to VC++ environment and using the debugger (1 week); 2) Introduction to Operating System Events (1 week); 3) Introduction to virtual devices (1 week); 4) Outputting text in a graphical

windows environment (1 week); 5) Introduction to user generated events: Scroll Bar messages (1 week); 6) Introduction to the windows Graphics Device Interface (1 week); 7) Processing input device events (1 week); 8) Processing timer events (1 week); 9) Introduction to message passing between parent and child windows (1 week); 10) Message passing between arbitrary window objects (1 week); 11) Threads (1 week); 12) Project: Putting it all together plus network programming (3 weeks);

Estimate CSAB Category Content in Credit Hours

	CORE	ADVANCED		CORE	ADVANCED
Data Structures	0		Computer Organization and Architecture	0	
Algorithms	0		Concepts of Programming Languages	1	
Software Design	2				

Oral and Written Communications - Every student is required to submit at least __0__ written reports (not including exams, tests, quizzes, or commented programs) of typically __0__ pages and to make __0__ oral presentations of typically __0__ minutes duration. Include only material that is graded for grammar, spelling, style, and so forth, as well as for technical content, completeness, and accuracy.

- none

Social and Ethical Issues - Please list the topics that address the social and ethical implications of computing covered in all course sections. Estimate the class time spent on each topic. In what ways are the students in this course graded on their understanding of these topics (e.g., test questions, essays, oral presentations, and so forth)?

- none

Theoretical Foundations - Please list the types of theoretical material covered, and estimate the time devoted to such coverage in contact (lecture and lab) hours.

- Concurrency control and mutual exclusion (5).

Problem Analysis - Please describe the problem analysis experiences common to all course sections.

- Given a problem description, determine the minimum set of events that must be processed to achieve the required functionality.

Solution Design - Please describe the design experiences common to all course sections.

- 1 object-oriented, event driven programming project (individual or 2 person teams, 4 weeks, design and implementation)

Other Course Information

- Additional Suggested Course Assignments
 - 1 midterm exam (50 minutes)
 - 1 final exam (120 minutes, around 70% programming)
- Planned Course Enhancements – Major Course Redesign

New Course Title - Net-centric Computing

New Catalog Description - Introduces the structure, implementation, and theoretical underpinnings of computer networking and the applications that have been enabled by that technology.

New Course Objectives – Students should be able to:

- Design, code, test, and debug simple event-driven programs that respond to user events.
- Develop code that responds to exception conditions raised during execution.

- Demonstrate a range of common networked applications including e-mail, telnet, ftp, newsgroups, and web browsers.
- Explain the hierarchical, layered structure of a typical network architecture.
- Summarize the many areas of interest that lie within the net-centric computing area.
- Discuss important network standards, including who develops them and how they evolve.
- Describe the responsibilities of the first four layers of the ISO reference model.
- Discuss the differences between circuit switching and packet switching along with the advantages and disadvantages of each.
- Explain how a network can detect and correct transmission errors.
- Illustrate how a packet is routed over the Internet.
- Install a simple network with two clients and a single server using standard host-configuration software tools such as DHCP.
- Illustrate how public-key cryptography works.
- Summarize various authentication protocols.
- Generate and distribute a PGP key pair and use the PGP package to send an encrypted e-mail message.
- Explain the different roles and responsibilities of clients and servers for a range of possible applications.
- Select a range of tools that will ensure an efficient approach to implementing various client-server possibilities.
- Design and build a simple interactive web-based application (e.g., a simple web form that collects information from the client and stores it in a file on the server).
- Illustrate how interactive client-server web applications of medium size can be built using different types of Web technologies.
- Demonstrate how to implement a database-driven web site works, explaining the relevant technologies involved in each tier of the architecture and the accompanying performance tradeoffs.
- Implement a distributed system using any two distributed object frameworks and compare them with regard to performance and security issues.
- Explain the issues for network management arising from a range of security threats, including viruses, worms, Trojan horses, and denial-of-service attacks
- Summarize the strengths and weaknesses associated with different approaches to security.
- Develop a strategy for ensuring appropriate levels of security in a system designed for a particular purpose.
- Implement a network firewall.
- Summarize the basic characteristics of sampling and quantization for digital representation.
- Select, giving reasons that are sensitive to the specific application and particular circumstances, the most appropriate compression techniques for text, audio, image, and video information.
- Explain the asymmetric property of compression and decompression algorithms.
- Illustrate the concept of run-length encoding.
- Illustrate how a program like the UNIX compress utility, which uses Huffman coding and the Ziv-Lempel algorithm, would compress a typical text file.
- Evaluate the potential of a computer system to host one of a range of possible multimedia applications, including an assessment of the requirements of multimedia systems on the underlying networking technology.
- Produce a description of the characteristics of a computer system (including identification of support tools and appropriate standards) that has to host the implementation of one of a range of possible multimedia applications.
- Demonstrate the ability to implement a multimedia application of modest size.
- Describe the main characteristics of mobile IP and explain how differs from IP with regard to mobility management and location management as well as performance.
- Illustrate (with home agents and foreign agents) how e-mail and other traffic is routed using mobile IP.
- Implement a simple application that relies on mobile and wireless data communications.

- Be aware of the many areas of interest that lie within this area, including networking, multimedia, wireless and mobile computing, and distributed computing.
- Outline the philosophy of object-oriented design and the concepts of encapsulation, subclassing, inheritance, and polymorphism.
- Design, code, test, and debug simple programs in an object-oriented programming language.
- Describe runtime implementation techniques for objects and classes.

New Syllabus

- Communication and networking: Network standards and standardization bodies; the ISO 7-layer reference model in general and its instantiation in TCP/IP; circuit switching and packet switching; streams and datagrams; physical layer networking concepts; data link layer concepts; Internetworking and routing; transport layer services
- The web as an example of client-server computing: Web technologies; characteristics of web servers; role of client computers; nature of the client-server relationship; web protocols; support tools for web-site creation and web management; developing Internet information servers; publishing information and applications
- Building web applications: Protocols at the application layer; principles of web engineering; database-driven web sites; remote procedure calls; lightweight distributed objects; the role of middleware; support tools; security issues in distributed object systems; enterprise-wide web-based applications
- Network management: Review of the issues of network management; issues for Internet service providers; security issues and firewalls; quality of service issues
- Compression and decompression: Review of basic data compression; audio compression and decompression; image compression and decompression; video compression and decompression; performance issues
- Multimedia data technologies: Review of multimedia technologies; multimedia standards; capacity planning and performance issues; input and output devices; MIDI keyboards, synthesizers; storage standards; multimedia servers and file systems; tools to support multimedia development
- Wireless and mobile computing: Overview of the history, evolution, and compatibility of wireless standards; the special problems of wireless and mobile computing; wireless local area networks and satellite-based networks; wireless local loops; mobile Internet protocol; mobile aware adaption; extending the client-server model to accommodate mobility; mobile data access; the software packages to support mobile and wireless computing; the role of middleware and support tools; performance issues; emerging technologies

Event-driven programming 3 hours

Introduction to net-centric computing 3 hours

Communication and networking 10 hours

Network security 5 hours

The web as an example of client-server computing 5 hours

Building web applications 12 hours

Network management 3 hours

Compression and decompression 5 hours

Multimedia data technologies 5 hours

Wireless and mobile computing 6 hours

Object-oriented programming 3 hours

Sample Project

In this project, the student will implement a simple Web browser that can perform the following functions:

1. Connect to a (perhaps instructor provided) web server.
2. Send HTTP commands to the server.
3. Download pages from the server, parse the HTTP commands in the page, and display the page's content.

4. Download and display the hypertext links from the server and implement the functionality of such links.
5. Download and display bitmapped images from the server.

This project requires that the student learn how to write reasonably sophisticated network programs using the Windows Winsock2 sockets library. It also requires that the student be able to download and appropriately handle different data types over the network connection (e.g. HTTP commands, bitmaps, sound files). It also requires that the student become familiar with basic parsing techniques, and the ability to handle (perhaps erroneous) input from the user in a reasonable way. It also requires the ability to design complex graphical objects using the Windows Graphics Device Interface.

Transition Plan:

It is anticipated that it will take three semesters to complete the transition from the current Windows programming focus to the net-centric material. In the first semester, the network programming aspect of the current curriculum will be enhanced and expanded. Additionally, the project will focus on developing tools that are enabled by the advances in network technologies such as the Web browser project discussed above. The second semester will add sampling theory, compression/decompression algorithms (for text, audio, image, and video information), network management and security issues, and wireless networks. The third semester will include all of the proposed modifications.