Course Number: CSC 642
Course Title: Human Computer Interaction
Number of Credits: 3
Schedule: Three hours of lecture/discussion per week.
Prerequisite: a grade of C or better in CSC 413 or consent of instructor.

Catalog Description

The design, implementation, and evaluation of human/computer interfaces. Topics include interface devices, interface metaphors, interaction styles, User Centered Design, testing, and quality assessment.

Expanded Description

The course has three main components:

- Part 1 introduces you to Visual Studio .NET and C# programming. This allows you to immediately begin building Windows GUI applications.
- Part 2 introduces you to user-centered design. You complete a PC-based Microsoft Windows project. The project allows you to put into practice concepts from User Centered Design.
- Part 3 introduces you to Pocket PC programming. The goal is to develop an application on a very different platform from a traditional Windows PC.

As one component of this course, you are required to learn the C# programming language. However, C# is virtually identical to Java, so there is no need to panic. As well, three weeks will be dedicated to covering C# concepts, so you should have a relatively painless introduction to the language.

Why C#? There are a number of reasons. First, C# is the most common Windows development language. The .NET API is gradually replacing Win32 as the de facto Windows programming application programming interface (API). Second, let's face it; if you are building a graphical user interface, chances are that your end user will be using Microsoft Windows. Without some knowledge of Microsoft Windows programming, and the design of GUIs, you can't program for 90% of computer users. Finally, it's just easier. While Java is nice, its UI features are somewhat antiquated. As well, Java GUI builders don't have the sophistication of those available for C#. You can spend several days coding a Java GUI, or about an hour coding the same GUI in C#. There is a reason more end-user, GUI-based applications exist for Windows than for any other operating system. Part of it is marketing and market penetration, but part of it is that it's simply easier to build GUI-centric applications for Windows.

Weeks 1 – 4	Introduction to C# and Visual Studio .NET	Lecture topics include C#, GUI building, event-driven programming, GUI widgets, speech recognition, text to speech output, custom rendering and drawing.
Weeks 5 - 11	User-centered design	Lecture topics include User centered design (UCD) specifically: Contextual Inquiry, Lo-Fi prototyping, Rapid prototyping, the design-prototype-evaluate loop in UCD, user testing and collecting feedback.
Week 11	Midterm exam	
Weeks 12 – 16	Developing GUI applications for pen-based entry on small screens.	Lecture topics include: 1 week of small screen/pen-input issues, then a research component including HCI projects at SFSU and at other universities.

As noted earlier, there are two projects in the course. The projects are done by groups of two or three students. Typical group size is three. Each group has a unique project they are developing.

Project topics are assigned as follows. During week three of the course, groups are formed. At that point, each group comes up with several project ideas, and each group meets with the professor for ½ hour outside class time. The project ideas are discussed, and we settle on a project proposal. Groups are then matched by the professor. Another group is assigned to develop the project proposed by your group, and your group is their client. At the same time, your group acts as developer for another group in the class. In this way, all groups are in dual roles of developer and client during each project.

Why this organization? First, the idea of everyone in the class developing the same project introduces a lack of collegiality. Students feel that their group has to do "as well" as other groups. I like to get away from this idea of competition in courses I teach, and instead encourage students to share code. It's also relatively hard to copy from another group when the other group is doing something completely different. Second, the project becomes a more real-world experience. You typically do not get to choose either your projects or your clients in the real world. Why should a course be any different? Finally, expressing your ideas clearly enough for another group is a learning experience. It teaches you why it's hard to get things right.

Software for the class

This course uses Visual Studio .NET and Microsoft Windows Professional. If database software is required, the recommended database is the Microsoft SQL Server Desktop Engine. As well, if web services are required, IIS running on Microsoft Windows XP Professional is recommended. Visual Studio, Microsoft Windows Professional, and all other course software is available free of charge to all enrolled students.

Course Objectives and Role in Program

With motorization of SW technology, the development of easy to use GUIs is becoming more an more important. This course is design to address these issues and it teaches you to build Windows applications with a graphical user interface (GUI). The goals of this course are to:

- Familiarize you with GUI builders, particularly Visual Studio .NET.
- Familiarize you with GUI widgets, including menus, buttons, toolbars, windows, etc., etc.
- Familiarize you with event driven programming.
- Familiarize you with the process of user centered design.
- Provide you with experience designing and building Windows applications.
- Provide you with some experience building Windows applications for non-PC platforms (specifically the Pocket PC).

Learning Outcomes

At the end of this course students will

- Understand principles of easy to use GUI design and event programming
- Understand method of User Centered Design as a preferred SW Engineering method for GUI design
- Be able to create GUIs using Microsoft technologies like C#
- Get basic experience in developing front end for Microsoft applications

- Apply object oriented programming principles effectively when developing small to medium sized projects
- Write robust code utilizing exception handling language features
- Use a code profiler to tune a program's performance

Method of Evaluation

Grading is based on a combination of individual and group project work. As well, a mid-term exam is administered in week 11. A break-down of grading is as follows:

Component	Weight
Individual assignments	25%
Project 1 (PC application)	25%
Project 2 (Pocket PC application)	25%
Midterm Exam	25%

Required Textbooks

Windows Forms Programming in C# by Chris Sells

Programming Microsoft Windows with C# by Charles Petzold

Recommended Reference

Modified by: E. Lank, D. Petkovic

Last Revision Approved: April 2007