ICS 105: Project in Human-Computer Interaction
Spring 2004

Instructor: Paul Dourish
jpd@ics.uci.edu

TA: Alex Baker
abaker@uci.edu

The topic of this class is not simply "user interfaces," but "human-computer interaction." What's the difference? User interface design concerns itself with the layout of information on the screen, with the selection, design, and placement of visual features such as information objects, interaction objects, and so forth. Interaction design concerns itself with the whole process of people interacting with computer systems. Frequently, this means looking beyond the "surface" layer of the user interface, and looking at the whole sequence of actions by someone becomes aware of opportunities for action, evaluates their potential effects, selects amongst them, enacts them, and monitors and interprets the consequences. Interaction design is concerned not simply with the design of the interface, but with the experience of using an interactive system.

Our central concentration will be with the interaction design process, and the central elements of that process are *design alternatives* and *iteration*. Design alternatives allow us to evaluate our designs more easily by comparing alternatives to each other; they prevent us from getting locked into a particular strategy too early. Design studies have shown that pretty much the only factor that always indicates better design outcomes is the number of design alternatives considered. Iteration means going through cycles of design, testing, evaluation, and redesign many times. The importance of iteration is that it allows us to test out and evaluate designs repeatedly, and detect problems faster. The faster you find a problem, the easier it is to eliminate it.

Of course, in a ten-week quarter, it's pretty hard to iterate with real software systems. However, interaction design involves a range of potential prototyping strategies that allow us to evaluate designs before we have real software systems built – walkthroughs, paper prototypes, etc. We will make use of these approaches to iteratively evaluate the systems that we will build in this class.

This quarter, the theme for the designs is social software. This is a very hot area of system development right now. Many new system and sites are being launched which attempt to capture and exploit the ways in which social groups are organized. The most obvious are the "social networking" sites such as Friendster, LinkedIn, Orkut, etc. In this offering of ICS 105, we will develop some alternative examples of "social software," using the social networking idea to support other kinds of software systems. We (Alex and I) will provide you with some basic infrastructure to support the social networking aspects, and you'll build the rest of the system around that. I'll provide some sample project ideas, although your own ideas are most welcome.

Programming in this class will be done in Java. You should know Java well by now, and we're not going to help you figure out language features. We will introduce you to Swing, the Java user interface toolkit, although you'll need to also do some background study in order to understand the nuances and the opportunities for applying it (this is something we'll help with.) You will build working systems that we can evaluate in the lab by the end of the quarter.

You will work in teams of four or five. Each member of the team will have particular responsibility for some part of the design process – design, evaluation, implementation, testing, etc – but everyone should be contributing throughout the process. Interaction design is always a multi-disciplinary, multi-perspective process, and the more people you have involved, the better it will work.

We will have to move pretty fast, especially at the start of the quarter, in order to get everything done. I want to have teams formed by the end of the first week, design sketches by the end of the second week, we'll do paper prototyping at the start of week five, and software prototype evaluation during week nine. This is a tough schedule.