

Sketchtools: Prototyping Physical Interaction

Matt Cottam

Tellart and Rhode Island School of Design
27 Sims Avenue, Providence, RI 02909 USA
matt@tellart.com

Abstract

Industrial designers working in traditional media have the luxury of sketching, playing, and experimenting with their materials before constructing a finished product. Designers working with electronics and computers are relatively impoverished. To “sketch” with electronics or computers would typically require extensive training in engineering and ready access to inexpensive parts—requirements that most designers can’t easily meet. This deficiency—this inability to work closely with materials before building with them—hampers designers’ efforts to make products sensitive to human use. This paper describes an attempt to address this problem in a human-computer interaction (HCI) design studio at a major design school. The course itself was an exercise in design: it worked within severe constraints to address a human need. We describe our attempt to shape the course to meet students’ most pressing needs; our students’ attempts to work within the constraints of the course; and the outcomes of the course for students and faculty. The paper suggests that the course offers one way to experiment with HCI concepts, produce innovative solutions to design problems, and—crucially—humanize new technologies and the design process.

1 Problem Statement

How can industrial design (ID) students in a human-computer interface design studio course use electronics and computers with the same speed, fluidity, and fearlessness afforded by less expensive and more flexible materials? How can the studio encourage students to be sensitive to human need—and to the complicated material and programmatic elements of their media? How can the studio create an environment in which students can use computer hardware and software as sketch materials? How can industrial designers extend their knowledge of engineering without encroaching on the work of engineers? How can an HCI studio balance an exploration of technology with a commitment to human-centered design? How can the course itself be approached as a design problem?

2 Background

2.1 Team Members

Matt Cottam, Nicholas Scappaticci, Brian Hinch, The Department of Industrial Design (ID) and The Digital Media Graduate Program (DM) at Rhode Island School of Design (RISD), RISD ID and DM Students

2.2 Project Dates and Duration

Courses: RISD ID: HCI Studio, 2001-2003 Fall Semesters; RISD DM: The Experimental Physical Interface, 2004 Spring Semester; RISD ID Simulation Studio: Combat Medic Training Simulator (COMETS) (with the Simulation Group at the Center for Integration of Medicine and Innovative Technology (CIMIT), Massachusetts General Hospital, Summer, Fall, and Spring Semesters 2004-2005; Umeå Institute of Design (Umea, Sweden): Tangible Interface Design, Winter 2005; RISD ID and DM: Microsoft Research: Design Expo Studio, Spring Semester 2005.

3 History and Context

The model for this studio curriculum was developed and tested in American and Swedish graduate and undergraduate departments of industrial design and digital media. Tellart, a design consultancy owned and operated by the faculty of this course, developed and donated many of the ideas and tools involved with the course.

4 Challenge

In the development of this studio course, the department, faculty, and students worked within numerous political, economic, intellectual, and technological constraints. These included:

- students with varying experience in digital technologies;
- the cost and complexity of electronics as raw materials for rapid and fluid design concept sketching;
- the learning curve involved with computer programming for the ID student developing HCI concepts;
- departmental budgets for computer hardware and software tools and materials;
- institutional control over computer labs, servers, software purchases, and installations;
- the environmental impact of sketching with plastic and silicon-based materials (sketch models are often modified and discarded as design concepts develop).

The challenge of the course was to work within these constraints and still humanize the educational experience, the design process, and the technologies used in the course.

5 Solution

5.1 Process

When we first contemplated the course, we started with what seemed a simple question: How can an ID student sketch efficiently and effectively for HCI concepts using computer hardware and software? Because we have taught the course several times, revising it each time, the shape and nature of the course have changed significantly. Still, we always worked within the rough parameters of the standard design workshop: class time was spent on demonstrations by faculty, field trips, guest lectures, discussions and critiques of exercises, short assignments, and a final project. Our task was to humanize the process—to turn our students' attention toward human need, and to allow them to experiment with their materials in a way that enhanced their understanding of HCI issues. In this section we'll describe the process of refining the conceptual design of the course; in the following section we'll describe in detail the ways in which we adapted our use of technology in the studio to meet students' needs.

In the first iteration of the course, we asked students to work in teams of three on a broad conceptual problem: Design a human-computer interaction about an unmediated human experience in nature. This approach—team-based and abstract—was designed to ease students into the course material, to encourage collaboration, and to prompt playful, creative experimentation.

What happened? The students struggled to balance what seemed to be competing needs: to work in conceptual terms and to design useful, human-centered products. Students were eager to make finished products—and voluntarily spent a good deal of money on materials with which to make them. (This proved to be doubly frustrating. Because they worked in teams, no one member could claim the final product.) But the problem we had given them was vague enough that they weren't able to make use of their existing knowledge or to explore more advanced ideas about human need, use value, and the manufacturing process. The nature of the assignment also came to frustrate the professors: our evaluations were necessarily more subjective than we had hoped. Finally, because students couldn't customize or miniaturize many of the electronics and hardware components, they designed installation-size projects (10'x10'x10')—an approach that absorbed too much studio space and forced them to begin the labor of production too early, cutting short their time for experimenting. We had, in fact, encouraged them to use installations; we thought that form would expand the range of input possibilities.

In the second iteration of the course, we asked students to work on a more concrete problem, and we did away with mandatory group work for the final project. Our assignment was: Design models and documentation for an interactive product that will serve a human need. We asked for looks-like and works-like models and Web-based documentation, and this proved to be a major advancement. Rather than trying to miniaturize electronics and code and build finished prototypes, students spent more time on research and experimentation. They focused on the scope of human factors, the capabilities of their technologies, and the challenge of communicating their concepts in a compelling, efficient form. The students seemed relieved not to have to worry about group dynamics. One significant problem: Because their solutions were so diverse, they couldn't take full advantage of our critiques; the projects didn't lend themselves to comparison.



Figure 1. Students acting out interaction scenarios with a “works-like” model

In the third iteration of the course, we asked students to design an interactive digital product for use by a human in an extreme environment—an even more specific task that led to more focused discussions and a concentrated use of materials. It also forced students to consider more carefully than before two key issues: human use and environmental conditions. Students also took advantage of inexpensive new computer input/output technologies, including software developed by our firm, Tellart, and hardware developed by Making Things. Now their material palette became expansive: they could work with analog sensory data input and analog voltage output. Rather than using switches to control content, they could use continuous analog sensing (sensors that track motion, touch, etc.). And with analog voltage output, they could control any electrical appliance—not simply audio and video content. We also encouraged them to consider networked products—products that can communicate with each other, and stable input controls that manage the content of Web pages and electrical appliances in various locations.

The course in its current form is the result of a series of conceptual, pedagogical, and technological refinements, each related to the other. A conceptual adjustment was necessarily a pedagogical adjustment, which was necessarily a technological adjustment. In a way, this may be one of the course's significant lessons: no idea, no action, no material exists in isolation. To conceive of a project, to study it, to work with its materials—these activities are parts of a whole.

5.2 Solution Details

In this section, we describe our use of technology in the course. As we suggested above, every technological refinement affected the conceptual design of the course, our pedagogical approach to the course, and our students' outcomes.

The first iteration of the studio experience began with an act of destruction: students took apart discarded off-the-shelf computer hardware—keyboards, mice, game controllers. This was also an act of creation: the electronic components became raw material for sketch-modeling.

Why use salvaged hardware? First, it allows greater freedom to experiment. The parts have been donated or found in the garbage; students are relieved of the stress of working with expensive materials. Second, it makes smart use of environmental and financial resources. Plastic and silicon parts are given at least one additional use and then recycled; and already limited course budgets and lab fees can be put toward field trips, books, and other materials. Third, it's an exceptionally flexible learning tool. An extracted keyboard controller can be quickly and easily modified to respond to an array of inexpensive industrial sensors and switches, as well as custom concept switches created by students. Simple input hardware—keyboards and mice—don't require special software drivers or computer ports, and they work with most computers and operating systems. Using standard hookup wire and simple soldering tools, students can incorporate controllers, sensors, and switches into models built in wood, metal, plastic, glass, textile, and other workshops. The models can then be connected to computer hardware via USB and can serve as multi-sensory input devices for Digital Signal Processing (DSP) and other software applications.

We also relied on software sketching materials. Students used Macromedia Flash MX2004 and ActionScript to capture data from input devices (keypresses and cursor coordinates), script for DSP, and author audio and visual interface and content elements. Flash files (SWFs) can be published on the Web and can link to other Web sites and datasources while receiving user input from an experimental physical interface; Flash can also be used to publish Web-based design documentation. Flash is commonly installed as core software at art and design schools; it has a vast user base and is the subject of a number of instructional books and Web sites. ActionScript is an accessible, multi-purpose, and powerful object-oriented scripting language—ideal for experimentation.



Figure 2. Modifying a keyboard controller (wired and wireless)

These were the tools we relied on in our original version of the course: sketch-modeling with discarded hardware, and software sketching with Flash. We chose them because they were accessible, easy to use, and environmentally sound—and because they allowed students to work directly with the media, forcing them to be sensitive to its possible use. These tools were helpful but limited. We could build multi-sensory physical input devices that would control content on a screen, in speakers, and on the Web—but the Flash Player limited the ways in which data could be received from and sent to hardware. Also, a modified keyboard allowed us only binary input (“A” key is “down,” or “A” key is “up”). We wanted our students to have more: more technological options, more possibilities as designers and thinkers.

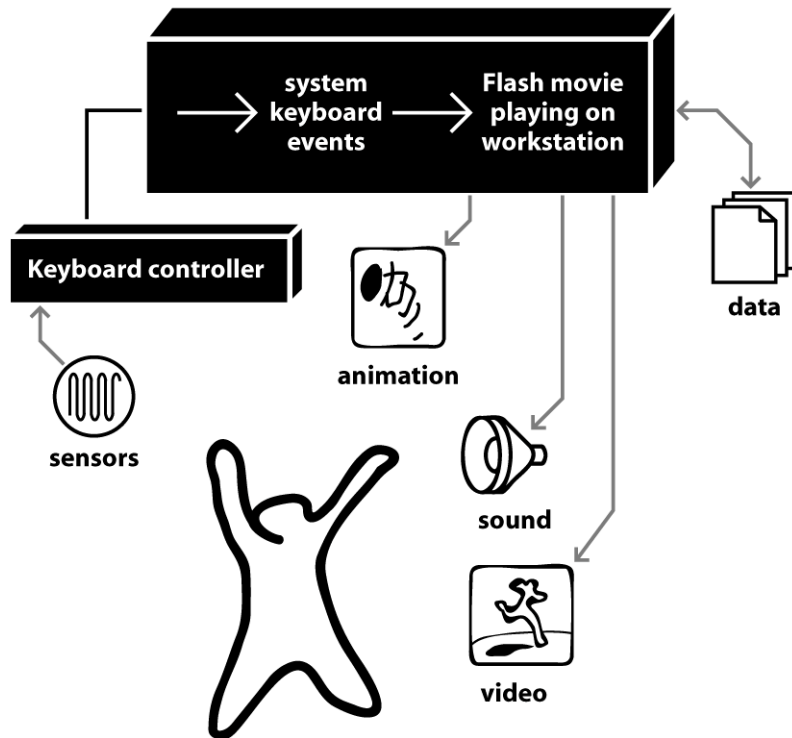


Figure 3. Hardware and software system using a modified keyboard controller

We started testing popular software solutions. Cycling 74’s Max/MSP allowed for greater flexibility in input and output data channels, and connected easily with popular analog and digital (A/D) converters (Teleo, iCube, etc.). But in the end, we stayed with Flash. It’s highly accessible to designers (Max—and especially MSP—was originally intended for computer music applications). It has a free and ubiquitous player and an internationally standardized object-oriented scripting language—a subset of ECMAScript (ISO/IEC 16262) and a good starting point for students eager to learn other programming and scripting languages. It can be used for drawing, typesetting, animating, controlling audio and video playback and effects, streaming audio, video, and vector graphics. It facilitates easy Web publishing; is easily integrated with Web applications; comes with a built-in XML object and socket; features pre-built components that allow for rapid prototyping; and, last but not least, it was already installed on our students’ laptops.

There was one major problem: How could we connect Flash with popular A/D converters? We worked with the staff of our consulting firm to explore solutions. Our first attempt was an elaborate system involving analog sensors; Teleo Analog In; Power and USB modules; a computer with Linux, Apache, JRUN and a Java-based XML server (FLOSC) installed; a Max Patch; and a Flash SWF that could send and receive XML and had some graphical buttons and sliders. Incredibly, it worked. At least it allowed us to control an animation on a Web site from an analog bend sensor, and to control an electrical motor by dragging a slider in the Flash movie. But it was an overly complex and inefficient composition of hardware and software elements.

Our next stage of development was to try and streamline the signal flow between analog sensors (bend, touch, heat, motion, proximity, etc.) and a Flash Movie, and between a Flash Movie and electrical devices (motors, lights, thermostats, heating elements, etc.). Our design consultancy built a platform-independent application in Java, called NADA, to allow more direct communication between A/D converters and Flash. This software is currently being used in the our courses, as well as at several schools internationally.

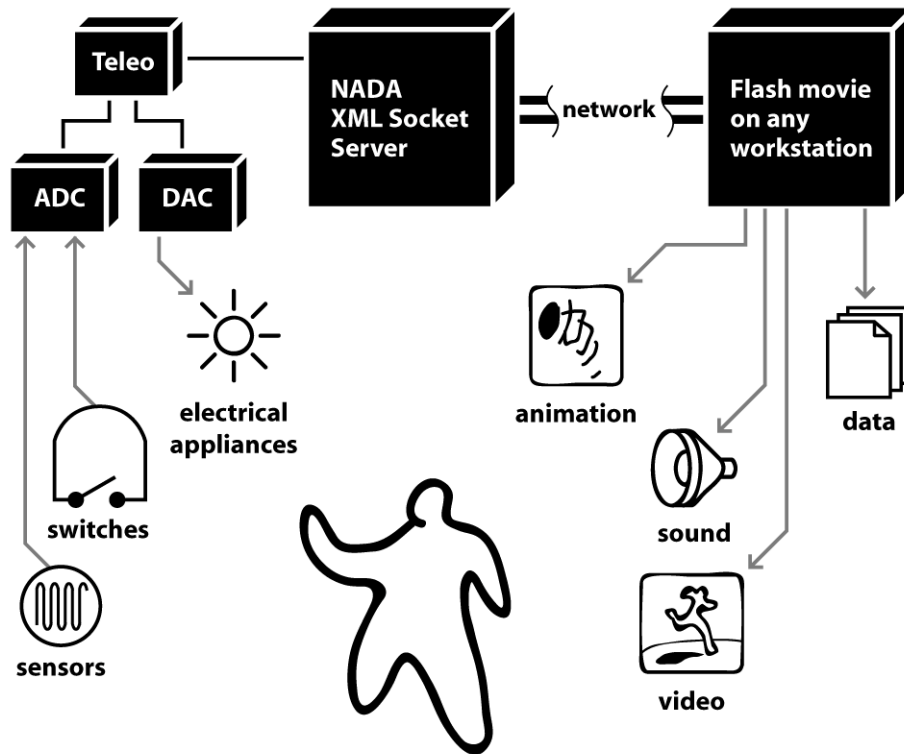


Figure 4. Hardware and software system using analog and digital converters

6 Results

Since the course was first offered, in 2001, we have seen a material change in its fortunes—and our own. The course has been over-enrolled; the waiting list exceeds three times the classroom capacity. Past students enroll in courses with the same faculty. A short video has been published about the course methodology. Other departments at RISD have agreed to partner with the industrial design department to offer a similar course. We have been invited to lecture at other institutions about the topics covered in the course. And Tellart has received contracts to apply the course's methods and tools to projects for clients.

None of these were explicit goals of the course; we didn't design the course in order to achieve them. But together they suggest that the course offers a meaningful, practical approach to HCI issues in a design studio—and beyond.

The goals of the course were to allow students to sketch in an unwieldy medium—the better to design solutions to specific human needs. In other words, we wanted to humanize the studio, the student, the technology, and the design process. In truth, our success is not easily measured. We can point to our students' projects, many of which showed a remarkable awareness of human use and environmental impact, or demonstrated exceptional skill in handling materials or developing concepts. And we can point to our course materials, which (we hope) became more helpful with each new session. But we would expect to have some talented students; and we would expect to provide every student with the most useful supporting materials.

Perhaps the best way to measure the outcome of the course is to talk in broad strokes about what we saw in the studio.

We saw students sketch with actual hardware and software—a process that allowed them to develop a vocabulary that bridged the gap between designers and computer engineers. We exposed them to specific technologies at specific times, with the aim of easing them into the science of their work. We developed demonstrations and exercises and assignments—using computer hardware and software—that followed a studio practice and a design process that they were already familiar with. We worked with each new technology ourselves, and we changed it to adapt to the needs of each new class of students. We made the engineering side of HCI design seem familiar, personal, possible.

We saw students design more robust products—because they understood their media at a deeper, more detailed level. They went beyond the color and shape of a plastic housing; they examined hardware components and software logic—enough to understand the qualities and capabilities of digital technologies. The most obviously useful outcome of this process was that they could design more sensitively shaped housings and human interfaces. They knew which questions to ask. Will the product require a lot of electrical power? Will it be heavy, or could it be light enough to be handheld? Will it cost a million dollars to make something that could reasonably be sold for one dollar? Will it require hard wiring to a wall outlet or to Ethernet? If it's wireless, what other components will be required? All of these questions affect the form and use of a product. If students have a solid understanding of them and the vocabulary that comes with them, they can more effectively communicate with engineers and develop more useful and robust products. Form really can follow function.

Finally, we saw students—equipped with a basic understanding of computer programming—research and design with a sensitivity to both sides of human-computer interaction: cognitive and computational. They learned basic principles like conditional logic: if this button is pressed, then turn on this light bulb; if this button is pressed in combination with another button, then play this video and turn on the light bulb. And that knowledge allowed them to design interactive scenes: when I input this information to the computer, it will interpret that information in this way and output this (these) response(s). Students wrote—as text narratives, not code—and physically acted out such scenes in class, a process that allowed them to develop concepts and make further discoveries through working and playing with coding sketch models.

Students, in the end, saw the design process more clearly. The course removed several layers of abstraction between process and product—between designing an interactive product and seeing it manifested as an object in a human environment. And it removed several dense layers of engineering study—technical knowledge that often prevents students from seeing clearly to the human experience of the product. Students explored the cognitive, social, technological, and economic factors involved in designing digital products. They did it through experience; they did it with a sound introduction to engineering and programming; and they did it, always, with an awareness of human need.

Tellart, through its Sketchtools Division (sketchtools.com), now offers NADA software and physical interface design workshops to schools and professionals. Our goal is to continue bridging design and engineering disciplines, and providing students and professionals with new means for thinking (sketching), designing and engaging their audiences with articulate and meaningful forms.

7 Acknowledgements

Special thanks to: Krzysztof Lenk who inspired me to teach; Mischa Schaub of Hyperwerk for first inspiring this work; Andreas Krach for helping me hack my first keyboard; RISD for encouraging and supporting this course development; Tellart for allowing us to do what we love at work; Nick Scappaticci—my trusty Tellart co-founder; Brian Hinch for leading the development of NADA and making the diagrams for this paper; Jasper Speicher for all the electronics lessons and teaching collaboration; Ryan Scott Bardsley for believing in the potential in bridging design and engineering and leading the COMETS project with RISD; Rob Morris for his editorial help in making this paper make sense; and all of our students for all of their bravery, work, and inspiration.

References

McCullough, M. (1996). Abstracting Craft: The Practiced Digital Hand, Reprint Edition. Cambridge: MIT Press.

Cooper, A. (1999). The Inmates Are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity, 1st Edition. Indianapolis: SAMS.

Laurel, B. (1990). The Art of Human-Computer Interface Design. Reading: Addison-Wesley Publishing Company.

Tilley, A.R., Henry Dreyfuss Associates (2001). The Measure of Man and Woman: Human Factors in Design. Indianapolis: John Wiley & Sons.

Sketchtools (NADA). <http://www.sketchtools.com>

Tellart. <http://www.tellart.com>

Rhode Island School of Design. <http://www.risd.edu>

Macromedia Flash MX2004 and ActionScript. <http://www.macromedia.com>

Making Things (Teleo). <http://www.makingthings.com>

Cycling 74. <http://www.cycling74.com>

Infusion Systems. <http://www.infusionsystems.com>

Open Sound Control (OSC). <http://www.cnmat.berkeley.edu/OpenSoundControl>

FLOSC. <http://www.benchun.net/flosc>

Umeå Institute of Design. <http://www.dh.umu.se>

Simulation Group, Center for Integration of Medicine and Innovative Technology, Massachusetts General Hospital.
<http://www.medicalsim.org>