DESIGN, MEET COMPUTATIONAL THINKING
AN INTERDISCIPLINARY EXERCISE IN DEVELOPING SMART PRODUCTS

Ed Dorsa, IDSA, School of Architecture + Design (dorsa@vt.edu); **Eloise Coupey**, PhD, Pamplin College of Business; **Tom Martin**, PhD, Bradley Department of Electrical and Computer Engineering; **Lisa McNair**, PhD, Engineering Education; **Jason Forsyth** (Ph.D. Student, ECE;) **Kahyun Kim** (Ph.D. Student, Industrial Systems Engineering.)

Virginia Polytechnic Institute and State University

INTRODUCTION

There is little argument today about the value of teaching interdisciplinary product development in industrial design education. It's the basis for some of the most widely recognized and respected graduate programs around – Stanford's d.school being perhaps the most heralded - and many undergraduate programs strive to provide this kind of opportunity at least once during their students' education.

Collaborative, interdisciplinary education prepares students for the professional product development environment that they will be entering after graduation; an environment in which team members are expected to contribute creatively across disciplines, to work with people whose experience and methodology are markedly different than their own to identify opportunities for new products and systems.

Often at an undergraduate level it is the Industrial Design program that advocates for and initiates these collaborations, usually with the Engineering and Business departments. Hence the curricular issues addressed and initial directions taken in interdisciplinary projects tend to have an Industrial Design bias.

But product development is a rapidly changing field; user needs and product capabilities are continually shifting even as we attempt to define them. Nowhere is this more evident than in the next generation of pervasive computing products and environments, where the very nature of the interaction is determined computationally. This new set of products, systems, and environments is driven by *"smart"* materials - *computationally malleable materials*[1] - that evolve throughout their life cycle, requiring design, engineering and manufacturing processes that are significantly different and more complex than current processes. Product development teams are entering a world where the look, feel and sound of an object will be dynamically determined by computation rather than by mechanics, and with which, at this time, most design students have little or no experience.

DESIGN THINKING

Each Fall Semester for the past six years at Virginia Tech, our interdisciplinary faculty group has taught a collaborative *Interdisciplinary Product Development Studio*, focused on pervasive computing products and services. Projects in this class have included ubiquitous, unobtrusive mobile healthcare devices; adaptive sustainable environments that support people as they age; and universal-design based interfaces for different abilities and increased connectedness. Involving students from Industrial Design, Marketing and Computer Engineering, the primary goal is to expose them to the value of interdisciplinary collaboration. However, each of these groups has already learned its own way of working, its own skill sets and has its own, unique, point of view on what product development "looks like."

Acknowledging the inherent bias of each of these disciplines, a basic tenet of the class has been to provide a working introduction to each profession, so that students can begin early to interact meaningfully with their studio partners. We begin by introducing the Marketing and Computer Engineering students to *Design Thinking.* When properly presented, Industrial Design's basic skill set tends to be accessible to the other fields, so that we can "level the playing field" a little. It enables the non-design students to feel more confident about their creative abilities, using intuition as well as the inductive and deductive reasoning they already know. We start with basic visual communication/sketching skills, introducing several fundamental drawing exercises, that they can use right then and that they can develop as the semester progresses.

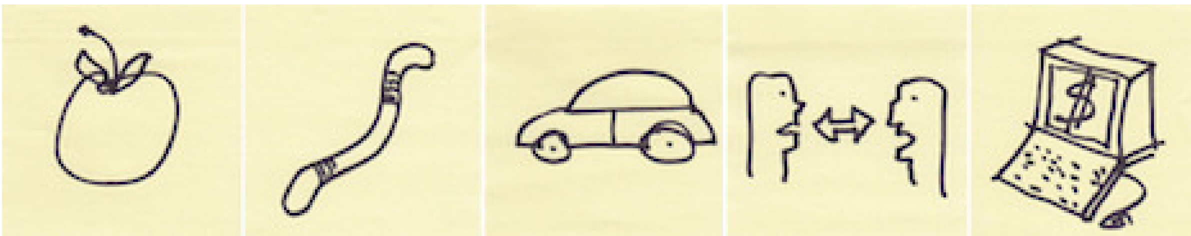

Fig. 1 Post-it note sketches depicting (L – R) an apple, a worm, a car, a conversation and e-commerce

We get them to quickly sketch out their concepts, to not worry about the quality of the drawings, and to use this new skill set to help them communicate ideas when words aren't enough. We also get them focused on the user as the central issue of the project. And, not incidentally, these exercises have also proven to be a good way to quickly build a collaborative environment that all the students feel they can contribute to.[3]

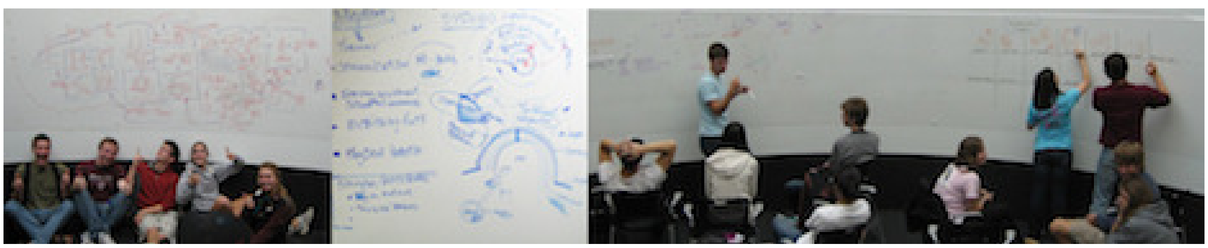

Fig. 2 ID, Marketing and CE students, from our Interdisciplinary Product Development class in a Design Thinking exercises.

But this raises an obvious question, "What do the design students get out of it?" They're already well versed in sketching and ideation techniques. Of course, they get to practice and develop their ability to work with other professions, which is valuable, but they're not learning any new skills through this exercise. In fact, the *Design Thinking* exercise can tend to reinforce the perception of them as the "sketching experts", and get them pigeon-holed as that being their primary, and only, area of expertise – if a team needs something "visualized" then they'll get the ID student to do it. From the beginning our goal with these interdisciplinary teams has been to give each student a sufficient understanding of what the other professions do to make it possible for them participate meaningfully in all decisions the group makes.

It became clear that we needed to develop a module on *Computational Thinking* [2] that would mirror, in terms of simplicity and basic learning, the *Design Thinking* module. We needed to provide the ID and Marketing students with a better understanding of computing, to bridge the gap that currently exists between design and technology.

COMPUTATIONAL THINKING

Our goal in the *Computational Thinking* module was twofold: one, that ID and Marketing students would see a new way of conceiving of products – an understanding that software is as much or more a part of today's products as hardware is - and two, that they would gain a new set of tools, both conceptual and physical, that they could use to expand and develop the capabilities of their product concepts. Ultimately, we hoped that the ID and Marketing students would realize that there was a new realm of product opportunities that they had previously seen as another profession's domain, but which they could now influence.

In our first efforts with this module we put together groups of students and had them work as a team to build interactive prototypes, in an hour and a half. We didn't try to tie this introductory exercise to the studio project, but instead gave them what we felt would be a manageable project given the short period of time they would have to work – in this case they were charged with designing a product or toy for a child.

We provided each group with an Arduino board, "an open-source electronics prototyping platform based on flexible, easy-to-use hardware…" [4] which allows users to control both input and output from the board; various input and output devices (input sensors for magnetism, force, proximity and light, and output devices such as servo motors, piezo electric speakers and LEDs) and some rough prototyping materials (boxes, post-it notepads, markers, etc.) Also, each group had at least one student from each of the majors, so there was somebody in each group who already knew how to code.


Fig. 3 ID, Marketing and CE students making interactive prototypes of children's products (L – R: a bank, a guitar and a light box.)

The result was, that while all of the students participated equally in conceiving what the product would do, and each had input on both desired inputs and outputs, only the computer engineering majors knew how to make it work – it was like each team had a chauffeur. They worked together to determine where they were going, but only the engineering majors could get them there. They enjoyed these forays into computing and it again was useful in building a collaborative environment. It was a step forward from where we had been, but it was still a long way from having ID and Marketing students understanding what was really required to make something interactive, and being able to accomplish it, even on a limited scale.

INTERACTIVE PROTOTYPES WORKSHOP

Most recently we took the *Computational Thinking* module to a new venue. We conducted a 2 ½ hour workshop on *Interactive Prototypes* during the 2012 IDSA Southern District Conference, hosted at the Interactive Product Design Laboratory at Georgia Tech,[5] for industrial design students and practitioners who were attending the conference. The workshop was led by Jason Forsyth, a Ph.D. student in Computer Engineering who's been involved for the past four years in our Interdisciplinary Product Development Studio, and who's dissertation work is focused on developing the intellectual tools to support the multiple viewpoints of interdisciplinary group work.

Again, we gave each group an Arduino board, an array of input and output devices and some physical prototyping materials. But this time each group downloaded a copy of Scratch, an open source programming language developed at MIT.[6] This particular version of Scratch, called Scratch for Arduino (S4A)[7], had been modified to work with Arduino and not just on a desktop computer. "Scratch is designed with learning and education in mind. As young people create and share projects in Scratch, they develop important design and problem-solving skills, learning how to think creatively, reason systematically, and work collaboratively." [8] It's a very visual language and has a "jig-saw puzzle" like interface. Since all the participants in this workshop were programming novices, we planned-in an early success to give a boost to everyone's confidence and to demonstrate that working with this software didn't require traditional programming skills. The Arduino has two LED lights hard-wired on it. One rapidly toggles on and off indicating that the Arduino and the computer are talking. The other is off. Our first exercise was to show the students how to program the Arduino to turn this light on. Everyone accomplished this in just a few minutes and there was noticeable excitement and satisfaction in the room early in the workshop.

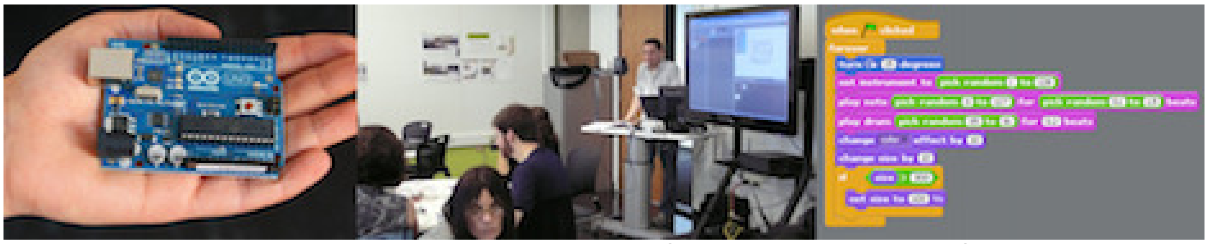

Fig. 4   L – R: an Arduino board; Jason Forsyth leading the Atlanta IDSA workshop; an example of a Scratch for Arduino programming "block."

We also provided a number of worksheets which demonstrated how to activate the various input and output devices provided. We went through about half of these so that everyone could gain some experience with Scratch. We left the remainder of the worksheets for the participants to use on their own project once they selected the inputs and outputs they wanted to use.

Fig. 5   Atlanta workshop attendees designing and building interactive prototypes.

Even given the normal confusion of getting a workshop started which reduced actual working time to something closer to two hours, this new combination of tools (hardware and software) was quickly incorporated by the participants, allowing each of them to begin applying a simple programming language and then for each team to make an interactive prototype. Many of the groups used the traditional industrial design process of iteration to make several prototypes, each an extension of the former.

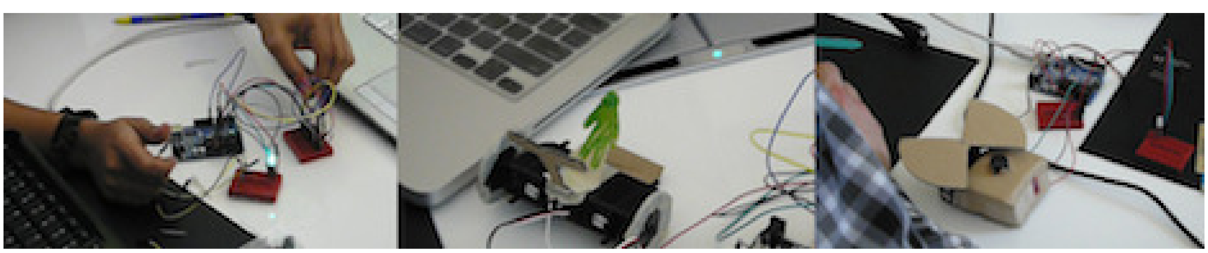

Fig. 6   Interactive prototypes using an Arduino boards, servo motors, proximity sensors, input devices and LED lights.

CONCLUSION & FUTURE DIRECTIONS

Creating interactive prototypes is a new experience for most design faculty and students. Its an ongoing experience in which we have worked to develop new techniques and methodologies with each iteration. To further our goal of computational thinking, we are looking at methods to incorporate storyboarding into the design and implementation of interactive products. Traditionally, storyboards are used in the early stages of design to illustrate particular uses, or scenarios involving the product. However, we would like to make storyboards a medium for implementing interactive prototypes, much in the way programming languages are currently used. We hope to develop a what-you-see-is-what-you-get approach to storyboarding such that it shows the behavior of the prototype, as well as the constraints of the underlying hardware. Since the desired behavior of an interactive prototype is so closely coupled with the particular hardware in use, the two must be considered at the same time. If we can reflect properties of the hardware in the storyboard, then implementation problems can be addressed early in the design and not after a prototype has been created; and ID students will have enough understanding and control to be fully involved in the design of interactive prototypes.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. EEC-0935103.

[1] Private communication with our colleagues Francis Quek and Ben Knapp.

[2] J. Wing, "Computational Thinking," Communications of the ACM vol. 49, no. 3, March 2006, pp. 33-35.

[3] Our sincere thanks to Chris Paccione and LUMA Institute for the use of several of the teaching/sketching techniques they developed for this segment.

[4] http://www.arduino.cc

[5] Our sincere thanks to Professor Jim Budd and the Industrial Design Program at Georgia Tech for allowing us to use the Interactive Product Design Laboratory for the Interactive Prototypes Workshop.

[6] http://scratch.mit.edu/

[7] http://seaside.citilab.eu/scratch/arduino

[8] http://info.scratch.mit.edu/Educators