

Social networks for lonely objects

John Anthony Kestner
Bachelor of Science in Design in Industrial Design,
Arizona State University, 2000

ARCHIVES

Submitted to the Program in Media Arts and Sciences, School of
Architecture and Planning in partial fulfillment of the
requirements for the degree of Master of Science in Media Arts
and Sciences, Massachusetts Institute of Technology, August 2010
[September 2010]

© 2010 Massachusetts Institute of Technology

Author


John Kestner
Program in Media Arts and Sciences
September 2010

Certified by

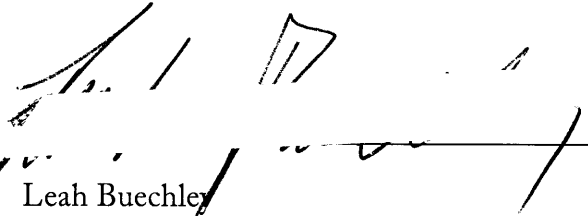

Henry Holtzman
Research Scientist, Chief Knowledge Officer

Accepted by


Pattie Maes
Associate Academic Head
Program in Media Arts and Sciences

Social networks *for lonely objects*

John Anthony Kestner

A handwritten signature in black ink, appearing to read 'Leah Buechley', written over a horizontal line.

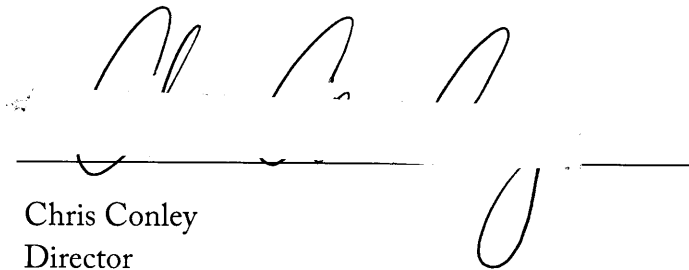
Thesis reader

Leah Buechley

Assistant Professor of Media Arts and Sciences

Social networks *for* lonely objects

John Anthony Kestner

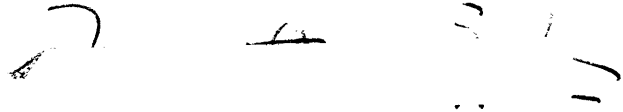
A handwritten signature in black ink, appearing to read 'Chris Conley', is written above a horizontal line. The signature is stylized and cursive.

Thesis reader

Chris Conley
Director
gravitytank

Social networks *for lonely objects*

John Anthony Kestner

A handwritten signature in black ink, appearing to read 'Hiroshi Ishii', positioned above a horizontal line.

Thesis reader

Hiroshi Ishii
Associate Director
Professor of Media Arts and Sciences

Acknowledgements

I thank my readers for their support and influence in my work leading up to this document. That goes double for Henry Holtzman, who has provided an amazing lens with which to focus my ideas.

I relied on the encouragement and feedback of my friends in the Information Ecology and Design Ecology groups. I thank them, along with friends in the IIT Institute of Design program and elsewhere, for our fruitful collaborations.

I deeply respect what my parents have given me. Their obsessions with systems and aesthetics, and a Montessori education, have borne fruit in this document.

And finally, I thank Julie because she continues to give me something to look forward to. I will never be a lonely object.

Table of contents

1. Introduction *15*

- 1.1. Motivation
- 1.2. Contributions
- 1.3. Organization of this paper

2. Context *19*

- 2.1. Invisible computing
- 2.2. Civility
 - 2.2.1. Everything is amazing, and nobody's happy: experience design in technology
 - 2.2.2. Emotional design for machines
 - 2.2.3. The burden of pushing/pixels: rich interaction design
- 2.3. Honesty
 - 2.3.1. Materials in electronic objects
 - 2.3.2. Gods or dogs: how smart should objects be?
 - 2.3.3. Craftsmanship of electronic tools
- 2.4. Physical object-oriented programming
 - 2.4.1. Sustainable device interactions
 - 2.4.2. Design for reconnection
 - 2.4.3. Prototyping electronic objects
- 2.5. Protocols
 - 2.5.1. Machine Protocols
 - 2.5.2. Human-Machine Protocols
 - 2.5.3. Machine use of Twitter

3. Exploration *51*

- 3.1. Presocial objects
 - 3.1.1. Kinesic Interface
 - 3.1.2. Watt Watchers
 - 3.1.3. Social Gardening
 - 3.1.4. Twitter Weather
- 3.2. Tools for designers
 - 3.2.1. Hand/Fingers

3.2.2. Virtual Social Objects

3.2.3. Baton

3.3. Social objects

3.3.1. AsDrawnOnTV

3.3.2. Proverbial Wallets: Tangible interface for financial awareness

3.3.3. Proximeter

3.3.4. Daydar: framework for socially motivated productivity

3.3.5. Connected dashboard

3.3.6. Tableau

4. Synthesis *93*

4.1. Social

4.1.1. Bus

4.1.2. Addressing & filtering

4.1.3. Discovery

4.1.4. Access Control

4.1.5. Persistence

4.1.6. Limitations

4.2. Super-mechanical

4.2.1. Evolutionary

4.2.2. Honest

4.2.3. Post-electronic

4.2.4. Nervous furniture

4.3. Social object apps

4.3.1. Independent living

4.3.2. Finance

4.3.3. Sharing memories

4.3.4. Waking up

4.4. Future work

5. Conclusion *113*

6. References *115*

1. Introduction

Now that microcontrollers have found their way into almost every household product, be it cookers, washing machines, cameras or audio equipment, a domain which was once considered pure industrial design is faced with many interaction design challenges.... Many interfaces of electronic products feel “stuck on”. This is not only a matter of form integration, but also a matter of how “display and push buttons” interfaces disrupt interaction flow, causing many electronic products to feel computeresque.

—Tom Djajadiningrat [23]

Visions of ubiquitous computing describe a network of devices that quietly supports human goals, but this may also add complexity to an already frustrating relationship between humans and their electronic objects. As we move from vision to reality, there is an opportunity to rethink how we interact with our objects and networks of objects. Exposing their inner workings through a social networking model can close the communication gap between man and machine, allowing users to take control of a tangle of interactions with many interconnected objects.

In this thesis I describe a series of design explorations that describe a new model for how consumer electronic objects should behave, centered around human functional and emotional needs. The objects are nodes in a network that collaborate to provide services. Each object is focused on input and output, with existing physical affordances overlaid with digital ones which are exposed on Twitter in a simple API. Apps in the cloud use the API to marshall the appropriate objects to execute human tasks. Using a social network as transport allows apps and their owners to manage a large network of computing objects with the same constructs that we use to manage many human relationships.

I intend that product designers take a leading role in the commercial realization of ubiquitous computing, through objects woven together into services. With a protocol for objects that

allows them to interact in a social network, designers can explore more sustainable lifespans for electronic objects, [13] decentralized “products” that evolve from simple rules applied to many objects, [7] and civil integrations of computing into our environments. [22]

The composer Claude Debussy is quoted as saying, “Music is the space between the notes.” There are countless songs in the connections between objects, waiting to be composed once the objects are freed from their factory-installed playlist.

1.1. Motivation

In spite of the increasingly digital character of our world, it remains physical, and in fact we are placing more emphasis on tangible interfaces for digital data. The physical object is gaining digital qualities, not the least of which is connectivity. As objects can be made to talk with other objects, whole new worlds of behavior open up. Ubiquitous computing applications variously involve objects that form ad-hoc sensor networks, objects that have a virtual presence and know their history in space and time, [2] or objects that work together to quietly manage the intricacies of our lives. [19,20]

But the complexity of interacting with networked objects and ever-expanding volumes of data threatens to overwhelm us, when we’re already managing an uneasy alliance between the physical and digital worlds in current consumer electronics. This is both a functional and emotional problem: the human interfaces of electronic objects are, as a whole, becoming more difficult to operate; [23] and we have created new forms of anxiety in trying to grasp invisible machinery. [3]

A way to resolve this is to make manifest the affordances to digital interactions, in order to manipulate and understand their workings. With the rise of the Information Age, we have been searching for a new balance between man and machine. As objects gain the gift of speech in order to talk to each other, we have an opportunity to converse with them ourselves. [4] Up to this point, those conversations are often tightly scripted by manufacturers. But just as the Internet has ushered in a new level of personal expression driven by users, there must be a similar medium for objects to communicate in ways limited only by end users’ imaginations. [11]

Bruce Sterling argues that successful products will have public application programming interfaces that allow them to be repurposed by the user in ways that the manufacturer could not foresee. [2]

This could be a competitive advantage to manufacturers—just like object-oriented programming allowed companies to bring products to market more rapidly, flexible systems of self-contained objects can allow companies to whip up every product customers could possibly desire without worrying about which ones will sell. A good ecosystem of expert users and apps will prevent that modularity from being overwhelming to consumers, and discourage companies from keeping the system closed, if they can be encouraged to open it in the first place.

I believe that invisible, ubiquitous computing in this spirit could be achieved with a variety of networked physical objects that serve as lightweight, specific interfaces to the user's cloud. These objects may not appear to be something new, but are simply more aware, connected versions of objects that we use every day—what Sterling calls *spimes*. The new functionality is folded invisibly into the existing tool.

1.2. Contributions

In this thesis I describe a series of design explorations and functioning prototypes that I have completed to research how future networked physical objects might interact with each other, and their owners. Through this work and its description in this thesis, I contribute and illustrate two sets of design principles, which I call the *super-mechanical* attributes of computational objects and *social objects*.

Super-mechanical attributes extend an object's familiar physical interfaces through electronic and computational components, in contrast to the often unintuitive operational paradigms currently employed by consumer electronics. Social objects are a new class of product; one that interoperates with other physical, software, and human entities using social networking services to manage system complexity. I demonstrate the viability of super-mechanical and social objects through functioning prototypes interconnected using the Twitter social networking service.

1.3. Organization of this paper

This thesis presents my vision of everyday objects, given social and super-mechanical properties, as a future building block of user-friendly ubiquitous computing. Beyond the introduction, the writing is divided into four chapters: Context, Exploration, Synthesis, and Conclusion:

Context looks at the state of electronic object-to-object and object-to-human interactions, and identifies gaps between ubiquitous computing and product design.

Exploration documents the design experiments I've built to close the gaps discussed in Context.

Synthesis reflects on the results of the experiments to define social and super-mechanical characteristics of future computational objects.

Conclusion lays out future directions for social objects and the object ecosystems within which they exist.

2. Context

This thesis proposes that developing ubiquitous computing from an idea to an everyday reality will require product designers. It follows that product designers will need help in negotiating the complexity of networked computing in order to do this. What follows is a discussion of challenges in designing human-electronic experiences that we face now, and in the future.

2.1. Invisible computing

Our computing experiences are dominated by activities such as social networking and information retrieval, consuming and sharing bite-sized chunks of content with little thought. The exponentially expanding flood of information from the Internet and sensors embedded in our lives is too large and varied to be represented in traditional formats, especially as screens have only shrunk. The cell phones that are currently at the center of this lifestyle are computers, after all, with an interface similarly dominated by screens, buttons and pointing devices.

Mark Weiser's vision of ubiquitous computing is a guiding light for my work, but I'm not alone. Dan Saffer remarks that "...Weiser's future is here. It's just in pieces, waiting to be loosely joined." [69] Characteristics of tabs, pads and boards [20] are reflected in today's consumer electronics, even if the networking glue is just coming into existence. But this idea of viewing and manipulating data using many tabs and pads is screen-centric. He acknowledged that this was not an optimal experience, noting that eyeglasses are an ideal tool because they do not demand your attention. They become invisible as you use them to see the world. [9]

So how do we draw the concept of invisible digital tools into the physical world? In this realm, Hiroshi Ishii's Tangible Bits provides grounding. Like him, I find powerful poetry in the affordances of old instruments, made rich by physicality and historical use. [21] This is in stark contrast to today's general computers. His work demonstrates how digital input and output can be integrated into physical objects, thus achieving a form of transparent computing.

While this describes the terminals of a ubiquitous computing environment, what happens in the spaces between? I found a compelling technical model in Open I/O and Pinkies, [14] which strongly influenced the Hand/Fingers platform for social objects. Pinkies are remote microcontrollers that use Ethernet ports to talk to each other over the Internet; Open I/O provides the glue: software running on a desktop on a Pinkie's local network, and a central server that allows the microcontrollers to discover each other. The Open I/O framework speaks to a larger need for standardized protocols by which devices can find and learn how to communicate with each other.

My advisor, Henry Holtzman, brings all of these ideas around invisible computing to bear in a domain that can make them literally ubiquitous: consumer electronics. He directs the CE 2.0 initiative at the Media Lab with Michael Bove, in which they propose a “new generation of consumer electronics that are highly connected, seamlessly interoperable, situation-aware, and radically simpler to use.” [12] This vision rejects the idea that more computing and connectivity means more complexity—instead, these enable cooperative systems that intelligently support human tasks without thought. [26] It is the lens that has brought everything I’ve worked on into focus.

All of these influences have been leavened by Dunne and Raby. Their body of work in critical design has been instrumental in exposing the sometimes inhumanity of technology, and convincing me that product design is the channel through which humane computing will propagate. This has been reinforced in my role as a designer, by interactions with many companies that produce consumer goods, including electronics, and the users and product designers I hope to empower. And so I answer Bruce Sterling’s call for designers to “design for, not just for objects or for people, but for the technosocial interactions that unite people and objects.” [2]

2.2. Civility

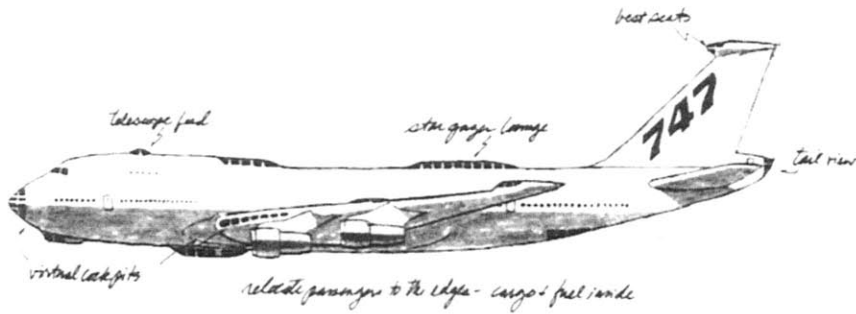
The first American astronauts had to persuade NASA to install a window in the early space capsules. Apparently to space engineers, the visual feast of seeing Mother Earth from space for the first time in human history was an unnecessary experience.

—Bill Stumpf [15]

This is a term borrowed from Bill Stumpf in “The Ice Palace That Melted Away.” He describes civility as the “something extra—the extra measure of grace—in the way we shape human behavior through objects and custom.”

If civility sounds like an old-fashioned word, maybe it’s because we don’t get enough of it from our technology. It doesn’t mean sharing pleasantries, tea and crumpets, though there is something great about insisting on those unnecessary formalities even in a harsh environment. There’s something great about demanding windows in a spaceship, too.

Generating civil things is the act of a designer. Civility is the humanity which we add to our works, for the sheer joy of it. The comfort of a lounge chair, the morale from wearing good clothes, the forgiveness of the Undo command, the community built from planting flowers in the front yard—these interactions lift our spirits. Who can’t use more of that?



*Redesign of the Boeing 747,
Bill Stumpf [15]*

2.2.1. Everything is amazing, and nobody's happy: experience design in technology

Everybody on every plane should just constantly be going, 'OH MY GOD! WOW!' You're flying! You're sitting in a chair in the sky.

—Louis C.K. on Late Night with Conan O'Brien [34]

This is an argument for quality over quantity. We tend to get caught up in counting bullet points when shopping for consumer electronics. It's to be expected—the current pace of technology drives features first, experience second. The joy, the art comes as time polishes products into something stable. Chairs or shoes all do pretty much the same thing, so they compete on emotional qualities. How do those shoes make you feel? Is that chair comfortable, and does it look good in the house?

Even as the underlying technology has constantly shifted, the everyday experience of electronic devices is stabilizing. It's not like the radical change in behavior that the telegraph brought. [37] Once we got used to that, the transition to voice or image was relatively easy. These were qualitative improvements to the essential quantitative breakthrough of communicating at light speed.

We're working through this in electronics. Capturing images was amazing. Megapixels are boring. Autofocus and digital "film" improved the experience of picture-taking greatly. Next should be bringing instant shutters back to consumer cameras [60] and making sharing as easy as a Polaroid. [61] The Eames of electronics, Apple is one consumer electronics company that seems to understand the fundamental human activities to support. [38]

Its reinvention of computers with touchscreen interfaces is the most recent evidence of this considered engineering.

It's a start. Unlike shoes and chairs, personal computers contain expensive technology, and so we make them chameleons. Now we have consumer electronics—cheap computers given specific roles. What's next? If we keep polishing, stripping and sanding down rough features, we might get many different objects each honed for a purpose, a satisfying extension of ourselves both functionally and emotionally.

In spite of the million-plus air miles I've flown over the years, I realized I had never experienced the thrill of flying....our technology has allowed us to separate the visceral reactions from the experience producing those reactions.

—Bill Stumpf [15]

So everything is amazing, and nobody's happy.

Bill Stumpf agrees with Louis C.K. that users take technology for granted, but plumbs deeper. He thinks that users are disenfranchised and disconnected, “reduced to little more than eggs in a carton” for their own safety. We have met the functional needs of the user while neglecting their emotional needs. Everyone should be amazed all the time, not by technology, but by the experiences it makes possible. Let's take a breather from adding features, and edit our human interfaces to make people happier.

2.2.2. Emotional design for machines

The transition from the Industrial Revolution to the Information Age (née Computer Age) is reflected in our products. As electronics are rapidly being developed and embedded into every aspect of life, their human interfaces have lagged. [23] There has not yet been much time, in the grand scheme of things, for the rough edges to wear down. The separation between function and interface has the potential to become more pronounced as products get networked. [2] Several designers have explored civil integration of computing into our objects and environments.

At the intersection of the industrial and interaction design disciplines, Tom Djajadiningrat is exploring aesthetics of behaviorally dynamic objects. [23] He has conducted exercises to discover taxonomies of expressive physical qualities. One of these had students create a pair of objects whose forms expressed two dynamic qualities (such as speed, weight, and accessibility) in common, and diverged on one. [Figure 1] He contrasts these possibilities with the interfaces of consumer electronics, stripped of symbolism and nuance in their generic binary controls.

Durrell Bishop sketches iconic interfaces in electronic objects. “Iconic” as in his exemplary Marble Answering Machine—but also in his use of abstract representations to get at the core concept of a complex system in a delightful way. [21] He has broken down anonymous plastic boxes into iconic forms that better represent their functions and create more meaningful experiences. The answering machine represents voice messages with physical marbles which can be placed on a spot to play back the message, or kept to play back at any time. He then extended this to demonstrate that we can assign any meaning to various tagged objects. [25] It’s through their rich physical attributes that we make associations with what they represent.

Tony Dunne and Fiona Raby have examined the deeper relationships between people and devices. They’ve done several probes into the tensions that electronic devices have introduced into modern culture, many of which seek to spark those feelings. [3] An archetypical example is the Placebo Project, [49] a series of critical objects that give form and character to ambient electromagnetic fields, to spur interactions with the inhabitants of the space, and observe the unintended feelings we develop about devices in our environment, and the living qualities we assign to inanimate objects. [Figure 2] They provide lessons on the anxiety that opaque machinery produces within us, but also how objects can feed our imaginations bits of narratives to manipulate our emotions for better or for worse.

Kelly Dobson’s Machine Therapy devices harness these emotions to create visceral feedback loops between the machines and people. She investigates how expressive interactions with machines can produce engaging and reflective experiences in humans. [4] Blendie is a kitchen blender whose speed is controlled by the pitch of the

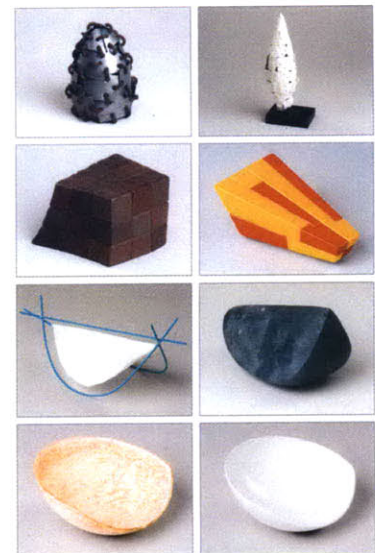


Figure 1. Opposite poles. Student exploration of physical expressiveness. Tom Djajadiningrat. [23]



Figure 2. Nipple Chair. The nipples vibrate in response to electromagnetic fields. Jason Evans.



Figure 3. Blendie. Pitch-activated appliance. Kelly Dobson. [4]

user's voice. [Figure 3] "A person may growl low pitch blender-like sounds to get it to spin slow (Blendie pitch and power matches the person) and the person can growl blender-style at higher pitches to speed up Blendie." The person is speaking the language of the machine, but the machine is meeting her halfway by understanding vocalizations. Omo, a soft melon-like machine, is a reflective object. It expands and contracts to match the holder's breathing rhythm, and sometimes guiding the holder to match its rhythm, encouraging self-awareness in the user.

Bill Gaver takes a similar approach in using machines as tools for reflection, but looks externally. In contrast to Dunne and Raby, his objects are primarily playful facilitators of connections with humans and their environment. [25] The Drift Table allows a user to browse an embedded aerial view of England by 'tilting' it with the placement of objects on its surface, The History Tablecloth "remembers" objects placed on it with an afterglow, highlighting past human activity.

2.2.3. The burden of pushing/pixels: rich interaction design

There is no longer any perceptually meaningful link between actions, form and feedback. Regardless of function, products feature the same 'display + push button' interfaces. These rely mainly on the users' cognitive skills, stretching their abilities to learn and remember.

—Tom Djajadiningrat [36]

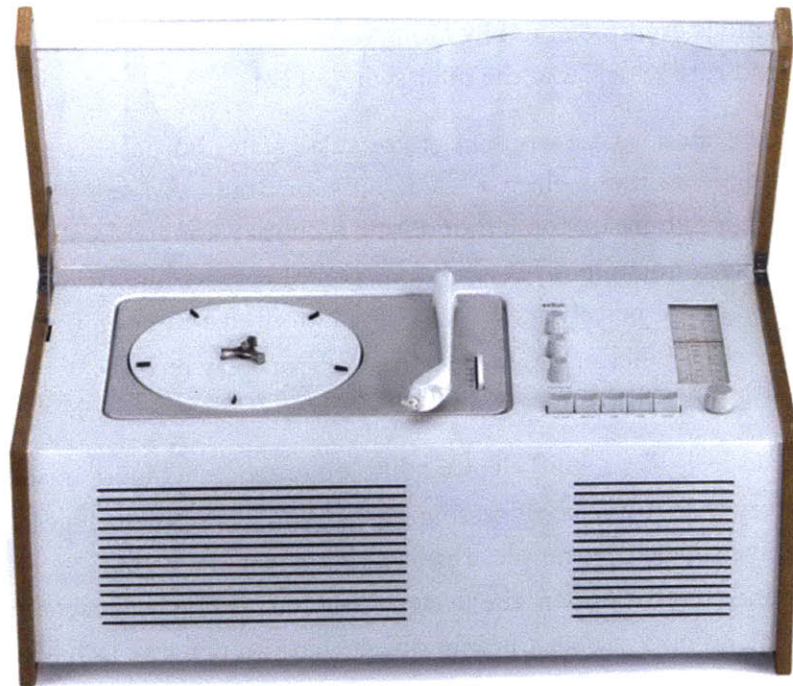
We experience much of the world through computers because they are extremely adept at connecting the scattered human colony. Most of the information computers deliver is visual, I suppose because we have well-developed visual cortexes. But we've got at least four other senses. Hell, we're covered in a touch sensor, and still barely any of our digital interfaces can be felt. Touch adds a critical dimension to human communication. Only the blind are ahead of the curve, and Braille "displays" prove that humans have a very capable sense of touch. Meanwhile, the role of touch in operating machines has been reduced to binary button pressing.

Objects in the physical world give us plenty of information which we absorb without thinking. That information is usually about the object itself, of course, but some objects relay abstract concepts just as pixels do. The size and heft of coins in our pocket gave us ambient information about how much we could buy. A grandfather clock gave us information about time, not just in the chimes every quarter hour but also in the rhythm of the pendulum marking the steady march forward.

Now our world is distributed across space and information must be virtualized to be delivered. Our money exists in a computer's memory in Delaware. We get the time from our devices that are synchronized to a time-keeping computer in Colorado. Pixels have taken over, but they're lossy—especially when there are so many that we have to tune them out. Ideal would be an interface that combines the manipulability of digital bits with the subtle richness of physical attributes, and managing that combination with grace.

Tom Djajadiningrat's quote above is from a section titled "The historical neglect of the body in interaction." Research by the likes of Durrell Bishop and Hiroshi Ishii has made progress in moving the needle back. But wiring explicit information to objects is only part of the interface. Electronic objects carry baggage that we deal with in the real world, things like durability, portability and power. To improve upon the whole computing experience, these must be addressed with both engineering and design prowess. So what comes next?

Braun SK4 record player, 1954. Introduced the transparent cover when other gramophones imitated furniture.



2.3. Honesty

Good design is honest. It does not make a product more innovative, powerful or valuable than it really is. It does not attempt to manipulate the consumer with promises that cannot be kept.

—Dieter Rams, via Vitsœ [58]

Honest design is a nebulous term. Dieter Rams doesn't elaborate, as is his style. His objects explain what he meant, but I can't get away with minimalist statements in a thesis. In this context, it's not about ethics. So what is an honest object?

It is what you think it is. Its form and materials indicate its nature. Well-defined affordances indicate functionality clearly. A streamlined car indicates its mobility; a streamlined pencil sharpener does not. Chrome is a fine finish for a metal bumper; plastic utensils with a chrome finish are a lie. [59]

It does what you think it does. Interactions are as much a part of honesty as materials are. Something that looks like a dial or button should turn or depress, and you should understand the effect of that action. Exposed hinges let you know how an object operates.

In a famous example noted by Don Norman, the handle or push plate indicate which way the door swings. [17]

Good design does use white lies to serve a greater purpose. Learning from Rams' electronics designs for Braun, Apple's product design has become monolithic (you may know it by its marketing term, unibody) to belie the complexity within. Because humans cannot perceive circuitry in operation, electronics must abstract their functions to "access the Web" or "play records," while retaining the transparency in interface.

The iPad feels like a solid element, like there's no machinery inside. The story that Apple is suggesting, that Braun electronics suggest, is that it really is that simple. This is a tightrope: The whole experience has to support the design story for it to be believable. That's all an honest object is—an object that you perceive as having a consistent model of interaction.



Figure 4. New iPod Touch (left), old iPod Touch.

2.3.1. Materials in electronic objects

Yet unlike my treasured box of lead type, I have no desire to hang on to all of the old computers I once loved. They lack the sensual attraction of the metal type, with its weight and the edges of its lead, the curves of its fonts, its ink stains, its missing letters and odd glyphs. There is something earthly about the metal type, something that deserves respect and that my old plastic bulky personal computers lack.

—Matt Cottam, *Wooden Logic* [73]

Touch is an underutilized sense in our electronics. [36] Our skin can feel heat or cold, rough or smooth, shape and plasticity. Touchscreens are all the rage, but they can't give the skin any feedback; they only do tactile input. Tactile experience is neglected even at a static level. Those easy-to-manufacture plastic cases crack instead of bending to our use—a binary sort of aging. They feature identically shaped buttons with identical stiff clicking action. We develop unspoken bonds with objects (and people) through touch memory. Soft worn leather shoes fit perfectly. A cold brass doorknob speaks of the weather outside, and of years of hands that have touched it. [Figure 5]

Industrial designers know and wield this, but in their race to the bottom, most consumer electronics are made out of ABS plastic. Apple has long stood out for its materials, most lately milled metal and glass. Designers will commend the use of elemental materials that have the right heft and are easily recycled. A brand new iPod Touch's chrome back is lovely. [Figure 4] It's a cool, worn pebble in the hand. On the other hand, the scratches on an older iPod Touch have gotten so dense that the chrome looks like brushed metal, but it's proven itself tough. There's little one can do to actually affect that "worn" pebble.

Apple carefully crafts all aspects of its products to reflect its brand message, from materials to advertising. Each new model, released like clockwork every year, is marketed to incite technolust. Does the speed at which it literally loses its shine play into this strategy? If so, it works—plenty of people are upgrading their hardware even though the iPhone is probably the most software-upgradable consumer electronics device ever.

Electronics we can cherish

The iPod Touch's emphasis on goal-oriented software over technical specifications makes technology less intimidating. But it also addresses the fear that something better is just around the corner. Largely, manufacturers have not completed this thought by making consumer electronics that look and feel better as they grow old with us. A recyclable aluminum shell is nice, but reducing consumption is a more direct path to sustainability. Can designers turn their skill in sparking infatuation to a more lasting love with electronic objects?

How do we give digital bits a form that we can bond with? Unlike my skin or an old dresser, [Figure 6] the history that my iPod Touch reflects in its shell doesn't seem meaningful. Is it because I use it for so many purposes that I don't associate it with any, or perhaps because I don't get too attached to an object with such a short lifespan? Maybe the industrial materials leave the impression of an alien-made machine rather than an old companion.

The Revo Heritage radio is notable as a mass-market product that presents a sympathetic philosophy. [Figure 7] The Rams-inspired aluminum-and-walnut shell tells you not to worry about what's inside, that it just plays music well. But that's a white lie—this box



Figure 5. Worn brass knob.



Figure 6. Hard Times Have Come and Gone, refinished dresser with original finish exposed in pattern. Rob Southcott.

is a computer with a wireless Internet connection that pulls audio from a spectrum of classic and modern sources: FM, DAB, iPod, Internet audio streams. That complexity underneath the hood does call for a display, which was carefully designed to disappear into an anonymous black rectangle when off. But will the physical object last longer than the digital object? It seems frustratingly closed for a device of its sophistication; the manufacturer has some ability to upgrade the software, and the user has none. [76] But the aesthetics set expectations of what it really is. This object is more future-proof than any other radio, with a form to match—and that, I can bond with.

Figure 7. Revo Heritage radio.





2.3.2. Gods or dogs: how smart should objects be?

In the TEI'10 keynote, John Frazer discussed his idea of actively evolving smart architecture. [35] An intelligent building might rearrange itself in response to environmental stimuli, he suggests, but also when it is bored (hopefully with serendipitous benefits to the inhabitants).

This scenario avoids a central pitfall of artificial intelligence systems—the expectation that they'll understand our commands. AIs tend to fail because we were promised a humanoid assistant that could parse natural language and intent, [47] and anything that requires us to adapt to *it* is disappointing. “Intelligent” implies a human-like interaction.

Figure 8. Roomba vacuum robot.



Figure 9. Hogwarts moving staircase in a Harry Potter video game. Warner Bros.

The bored building has a mind of its own, and while it may take the inhabitants' input into account, it would not be expected to respond directly. Instead of a servant, the user might see it as a god—an entity that acts in grand and mysterious ways, [Figure 9] but one that you can try requesting favors of.

The Roomba presents the opposite approach. [Figure 8] Unlike androids which fall into the uncanny valley trying to imitate us, it's just a low, round thing. In its bumbling motion and R2-D2 beeps, it has the character of a pet. [43] A dog doesn't understand much of what you say, but it's forgivable because it's trying to please you. Clearly there's room to tolerate error if you've got the right presentation.

This is difficult. It's not just about appearance. The Roomba doesn't look at all like an animal. And when attempts at lifelike companions fail, it's because they are not honest. They're teddy bears that are rigid and immobile. Or they're humanoids that cannot understand human communication. These are trying to be something they're not.

Why do objects need to be “smart” at all? There is a third way, in which they are tools that don't call attention to themselves. Their purpose is to make the user feel smart, or strong, or skilled. They don't need an artificial personality, because they reflect their owner's personality.

An honest tool, smart or not, has a consistent presentation from form to interaction. It should look like what it does, and do what you'd think it would from looking at it. That's a tricky thing with objects that have both physical and electronic affordances. But like any interface, it should be as transparent to its function as possible, and pleasantly exceed expectations. This requires craftsmanship.

2.3.3. Craftsmanship of electronic tools

Brenda Garand, the Dartmouth professor, picked up a C-clamp to which Fane had welded a pair of angle brackets. She said, "This is something that Larry made for a specific purpose. It was for holding on to something at multiple angles."

Jim Osman laughed, and shook his head. "That is so Larry," he said.

"That's the power of the object," Garand said. "It's the history, it's the touch, it's what is left of that person."

—David Owen, "Tools" [72]

Astronomy, driven by the needs of explorers, has left us instruments that were so beautifully crafted that they remain in our cultural consciousness long after they have been obsoleted. These measured, calculated and modeled a universe of which we were the center. For instance, the astrolabe [Figure 10] was typically brass and circular with the Earth at the center; it was useful for measuring time, heights of celestial bodies and buildings, from the latitude for which the instrument was designed.

The complexity of computers is why Bill Buxton called for a divergence toward a multitude of simpler tools, looking to the design arts field for leadership. [19] David Rose describes his father's interaction with his barometer. He shuffles into the hallway to tap it every morning, which makes the stiff needle move in the direction of the changing pressure, grunts acknowledgement, and shuffles out. Rose sees a future of "beautiful antiques" resembling that instrument scattered around the house, each providing rich, specific data in an elegant tool. [55]



Figure 10. A 14th-century astrolabe, used to measure time or distances. The Whipple Museum, University of Cambridge.

That's why it's so important that the leadership be craftsmen... you gain something in the heart.

—Søren Petersen, manufacturer of the Chair. [10]



Figure 11. Hans Wegner's Round Chair, or the Chair. Dansk Møbelkunst.

Hans Wegner made furniture. His essential design is known simply as the Chair. [Figure 11] Its oak pieces are carved and sanded and fitted together by hand to make a seamless functional sculpture. Wegner designed the Chair to last at least 50 years (a good thing, since it costs \$4000). It's finished only with soap flakes to allow it to develop a patina from long-term use. [10] The craftsman tells a story of humans through his object—the ones who made it and the ones who owned it.

It's fine that Ikea can make affordable and pretty, if not durable, chairs. But these draw on the soul of greater furniture created by craftsmen. Most industries have such leaders, from fashion to food. This is difficult to find in electronic products because the craftsmen have to understand software and hardware and materials and interaction. [19] They need more accessible tools and skilled assistants.

The electronics industry is missing the spiritual equivalent of the Chair. Given the support, what could craftsmen do? They might turn the feature-driven electronics industry on its ear, and show it how to use the technology in service of a human experience.

2.4. Physical object-oriented programming

When we were an agrarian nation, all cars were trucks. But as people moved more towards urban centers, people started to get into cars. I think PCs are going to be like trucks.

—Steve Jobs [85]

The opacity of new computing devices is approaching that of consumer electronics. As the network takes prominence, the computer is just a discrete component of that system. Hacking it will be about as relevant as modifying an IC. In the long view, abstraction away from the parts will allow us to create more complex, interesting things through connections between those components. Witness the iTunes ecosystem, or Twitter. They're not perfect, but shoveling data around in the background allows us to focus on a richer experience that wasn't possible before.

Tinkerers have cause to be worried. Many geeks will protest that they learned technical skills by diving into the guts of computers, and this is no longer possible if the guts are locked away. Perhaps a happy medium can be struck by providing a simple locked-down interface by default, along with a way to get under the hood that limits possible damage. Twitter does this by offering a web-based client and an API that developers can use to create varied and powerful interfaces to Twitter's ecosystem. The app model gaining popularity with smartphones encapsulates functionality in a way that it can easily be added and removed by the user, but depending on the platform, deeper access can also be gained by expert users.

But openness is often at odds with simplicity. Periodically, complexity in computers is compressed down into a foundation that allows increases in programmer productivity. It hasn't stopped programmers from complaining as the world moved from assembly language to C to compiled object-oriented languages and interpreted languages, but each of these has marked increases in the number and efficiency of coders.

The parts of what constitutes a computer are just shifting around. Even physically—input and output will be wherever your hands or

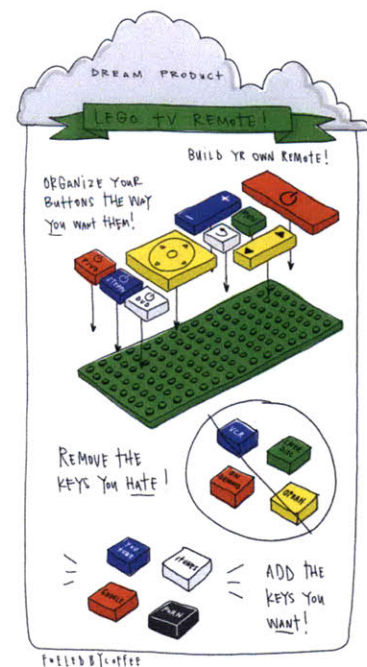


Figure 12. Modular TV remote concept. Craighton Berman.

eyes are, and storage and logic wherever you like. [41] The model of object-oriented programming on the network scale makes sense for physically separate parts of a single logical program. We will encapsulate the entire software stack of an electronic device in a single software object that passes messages over the network to other physical/digital objects. The important thing is that we get good APIs to wire together these pieces as we see fit [Figure 12] and change out components, as in a home theater. Music awaits in the connections between.

2.4.1. Sustainable device interactions

We are lacking tools to control how we interact with the piles of objects that make up our environment. Those tools must be simple, since complexity is what needs to be eliminated in the first place. Who will build such objects? Erik von Hippel argues that giving customers tools to co-design products can give companies a cheaper, field-tested development process. [5] The iPhone ecosystem is co-designed in that the customer selects the apps that create their desired experience. It's a generic consumer electronics box, mass customized.

Reduce (complexity)

Per the 80/20 rule, most people use only a subset of all of the core functions of these general-purpose computers. My iPod Touch is mostly just a note-taker and -sender; your iPhone may be a social network wrangler (by voice, word or photo); you can give it to someone else and they will reconfigure it to be a media console. These are different from computers precisely because the single-tasking and sandboxes allow users to customize without feeling that they're breaking something else. When I touch an app icon, it fills the screen, and I'm holding Facebook or an atlas. Unlike a computer, nothing you can touch on the screen will disturb this—only the physical 'home' button can.

Reuse (components)

As current consumer electronics hardware gets cheap on the used market, won't we just run single apps on a handful of iPhone devices? One is the universal remote, another the home stereo, this is the car GPS, that's the recipe catalog. It might be enabled simply

by attaching the right piece of hardware, like the Motorola Droid smartphone's car and bedside docks or the various holder that turn a WiiMote into a steering wheel or a tennis racket.

In a previous section, I discussed how the materials of electronic objects often go hand-in-hand with the functional life of those objects. The physical objects tend to be cheap and flimsy, and the software functions become obsolete as quickly. If designers do implement electronics that can develop a patina, the functionality also needs to age gracefully. By its software nature, this means openness and upgradability.

Reware [33] is an early attempt at this. The project proposes to update the firmware of old consumer electronics to turn them into general purpose computing blocks, but is currently raw and nerdy. Ideal would be a system that could just ask you what aging electronics you have and give you "apps" that just wipe them out and install the software to do the desired task.

Recycle (computation)

Computation is cheap enough that it's rarely worth it to try to crack the nut presented by closed devices as Reware attempts. But the app store model has changed consumer perception of electronics by exposing a malleable software layer, and static functionality is increasingly unacceptable. The fluctuating requirements of the user must be reflected by flexible computers. Convenience is driving this change. Computation may be cheap, but time and space is not.

2.4.2. Design for reconnection

We tend to think that consumerism is about loving our stuff. But it's not. Consumerism is a result of not loving our stuff at all. In fact, we have so little regard or respect for our material goods that we dispose and replace them with ever increasing regularity.

—James Shelley, "Love Your Stuff" [62]

Designers have an opportunity, even responsibility, to influence the product's whole life. While the following approaches specifically

discuss manufacturing physical objects, which is outside the scope of this thesis, they suggest a philosophy that electronic objects can pick up first—design for reassembly. As more of our products’ functionality is defined in code and enabled by the network, we don’t have to disassemble objects to give them new life. Design for reconnection means that products can find new life by loading new software and reconfiguring the network. But it will require open, flexible connections. This is a goal of the thesis work.

Design for reassembly

Design for disassembly is an element of the cradle-to-cradle design movement that considers what happens to a designed object when its useful life is over. [42] We generate tons of e-waste, and waste in general, faster than we can break it down. The efficiency of manufacturing processes is a marvel of modern technology, but the drive to make things smaller and seamless results in products that are harder to break down. Design for disassembly makes it more likely that we can recycle parts, or repair and repurpose objects to give them a longer life. [78]

Products with APIs can help here. Instead of breaking products down into components, electronic devices themselves can become the components, following an object-oriented model. After all, OOP promotes maintainability. Following are three ways to implement a “design for reassembly” strategy for physical products.

Start at the top: manufacturers

Of course, selling companies on maintainability and reuse, whether physical or digital, has always been a challenge. In the small view at least, they face public pressure to be good corporate citizens, and legal obligations to meet in recycling and waste disposal.

It could also be sold as a competitive advantage. Just as object-oriented programming allowed companies to bring products to market more rapidly, flexible systems of self-contained objects can allow companies to whip up every product a customer could possibly desire without worrying about which ones will sell.

An ecosystem of users, instructions, and whatever the physical equivalent of install scripts is, will prevent that modularity from being overwhelming to consumers, and discourage companies from

keeping the system closed...if they can be encouraged to open it in the first place. Online communities of users have popped up around Ikea's products, sharing instructions for building their own objects out of the relatively modular furniture, while Ikea itself still maintains a brand and product line for the less imaginative. [79]

Start at the bottom: tools for end users

Salvage, a project by my colleague Greg Elliott, is a workbench for cataloging the components of unwanted objects. [80] It could draw on selfishly motivated data sources. I could use it to catalog my own electronics in the way that proud collectors of all stripes do. When I realize that I'm never going to use that old phone again, it'll be ready—I flip a switch and all the parts are up for grab. Or people will make offers even before I consider it salvage.

This could be combined with collections of dynamic assembly instructions. Instructables is an online community which shares directions on how to do or make almost anything. [81] A system building on this might discover what components are available, and provide instructions to make a number of different new objects.

Start over: create the future of manufacturing

David Carr's Mantis "home fabratory," another project in my research group, is a sub-\$100 3D printer. [82] Being an order of magnitude cheaper than most other CNC milling machines, it projects a near-future where users share the responsibility of designing one-off products. With the barrier of cost knocked down, the software that will be used to create new objects will be key to unlocking this potential. 3D modeling tools will have to adapt, and at the same time, get much, much easier to use.

There are some alternate approaches to creating whole new objects in a traditional CAD program. Most people will only need to modify existing objects. Carr and Sean Follmer have developed CopyCAD, which allows the user to "draw" instructions on parts (using a projector and computer vision) that the machine can use to modify them, or print new parts with the changes. [83] It could be a companion tool for Salvage or other libraries of parts, modifying components as necessary to make the best use of them. Imagine model files that can dynamically change to absorb whatever parts you've got on hand.

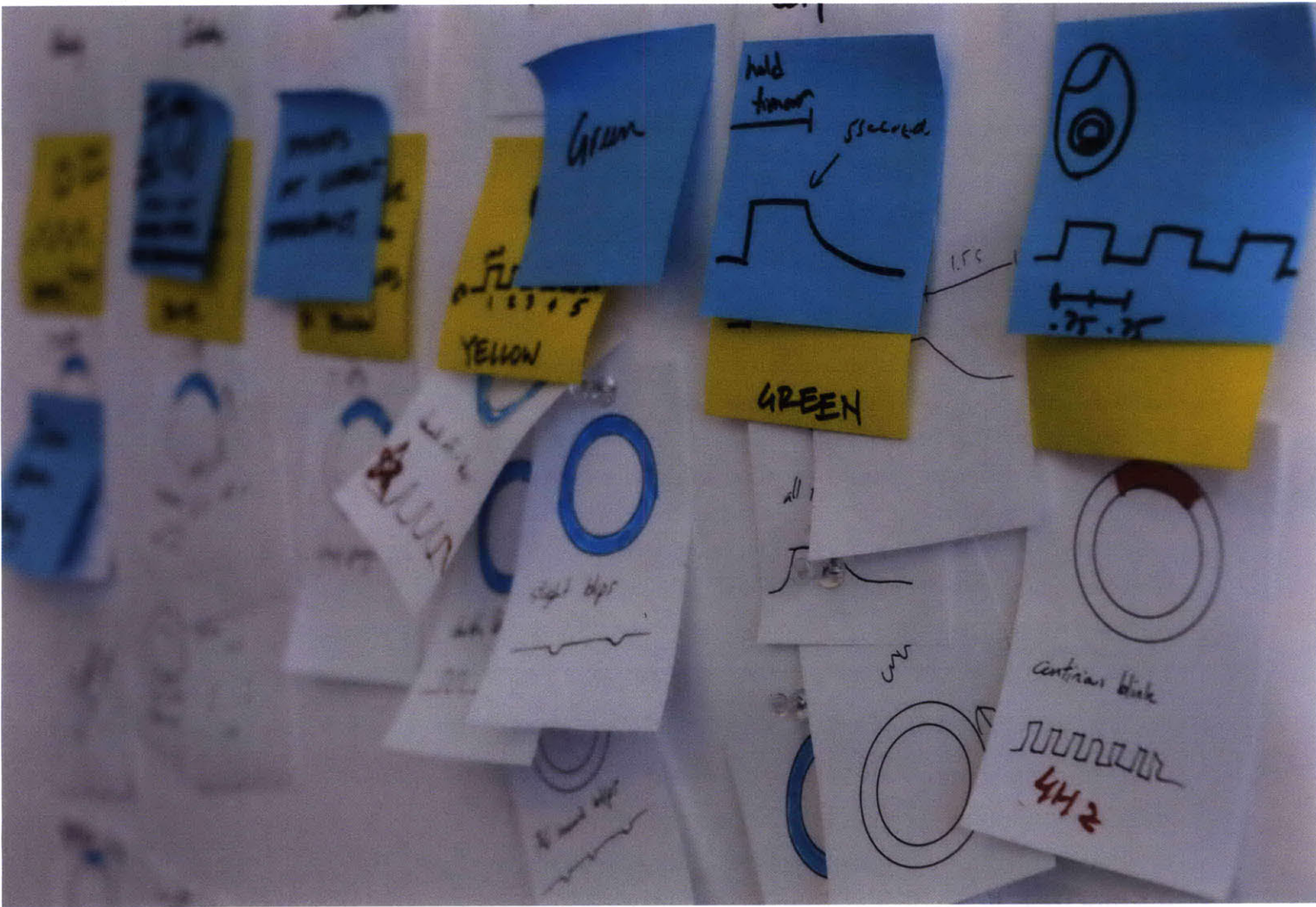


Figure 13. Interaction sketches for an electronic device. Coulter Lewis.

2.4.3. Prototyping electronic objects

Over the years I've seen interaction design thrive while the industrial design profession has gone into decline. While the physical object is essential ... it is the larger experience with these products that is most meaningful to the people who use them.... The time has come for industrial designers to redefine their profession.

—Ken Fry [24]

Product design studies all sorts of relationships between humans and objects—is it still relevant as interactions with multiple electronic objects take the foreground? Djajadiningrat [23] notes that the position that product designers often find themselves in now is to wrap existing electronics in a shell and stick on an interface after the fact. And now, simply addressing the physical

interface is not enough. Products have broken out of physical shells to become services hosted across an ecosystem of electronic objects. [1]

Product designers are recognizing that the profession must adapt to meet these challenges. [24] Designers' core skills involve solving problems and communicating ideas through prototyping. Designers use prototypes to spur discussion with each other, clients and users. So how do designers translate ideas for electronic objects into interactions that people can react to? Looking even further, is it possible for designers to prototype these interactions across networked physical objects? The appropriate tools can remove technical constraints, freeing them to focus on developing more humane and innovative object interactions.

2.4.3.1. Existing tools

Several physical computation tools useful for prototyping electronic interactions have already been developed, but simple networking capabilities are out of reach for designers at this point. Following are several of the most relevant tools to product designers.

Phidgets [8] is an early toolkit which began with the model of making physical versions of GUI elements, such as knobs, sliders, motors and other sensors and actuators that are attached to a computer controlling them. The mental model indicates that the audience would be software developers who want to expand their reach into tangible interfaces, and indeed, Phidgets are programmed with behavior using code libraries for many different programming languages.

More directly targeted at product designers and based on research in the field, d.tools [51] takes a different approach. Finding that designers are unsatisfied with interaction prototyping tools and the inefficient iteration process involving technical specialists, the authors designed an integrated suite of tools for physical prototyping that support iteration not only of prototyping, but of testing and analyzing designs. It uses a visual dataflow programming style to program behaviors based on the thesis that textual programming languages are too difficult and slow for prototyping designers. Aside from the monolithic approach, it has

similar limitations to Phidgets. And the authors' interests are perhaps reflected not only in the HCI-oriented testing and analysis tools, but in the scenarios used for evaluation, in which participants were asked to prototype typical consumer electronics.

LittleBits [57] is another toolkit targeted at designers, one which does not require the user to program behavior into software. It contributes an original interaction model that makes prototyping approachable, connecting discrete components with magnets. Behavior is determined by the arrangement of the physical components. It does not offer complex interactions or networking.

Arduino [16] is a more general toolkit, and is already a popular platform for physical computing that is in use by a sizable community including artists, hobbyists, researchers and designers. It does not have a standard way to communicate with other objects beyond a low-level serial library, and requires coding in a dialect of the C programming language but provides a simple starting point. It also requires some knowledge of electronics, not providing much in the way of built-in sensors and actuators. The Firmata library [70] provides low-level communication with a host computer, so that code is written for the computer instead of the Arduino, which remains tethered to the host.

2.4.3.2. Research of designer processes

Notwithstanding the tools described here, observations and interviews I've done with practicing product designers indicate that they have not found appropriate tools to prototype electronic objects. [56] Designers are communicators. This is why they sketch 2D and 3D forms—in order to communicate ideas to clients or collaborators. With the time dimension and decision trees of interaction added, this becomes a more difficult task. Programmable microcontrollers with a gentle learning curve such as Arduino are currently the best way to “sketch” electronics.

I observed that designers and electronic engineers working on a project together communicated through hand gestures and doodles on a board of sticky notes. In Figure 13, these techniques are used to describe the characteristics of animations that play on a ring of LEDs as feedback to the user of an electronic device. Describing the end experience is more natural to human designers than

describing the algorithms that generate output, as coding often forces us to do, especially when trying to create smooth “analog” feedback.

While some designers are familiar with code, primarily in the form of ActionScript, it can be difficult to capture expressive input and output with a textual programming language. Hartmann, et al. make a case for a programming-by-demonstration approach with Exemplar [87], though it focuses on translating sensor data into recognizable events that a program can act upon—input rather than output.

I believe that simple tools to capture and digitize expression can allow designers to focus on the quality of that feedback rather than the implementation. These should take advantage of the fine motor control in our hands for input. A gesture or a drawing could be translated into a pulse pattern for a vibrating motor or the motion of an on-screen animation. Or an alarm clock can take on the qualities of a doorbell that make it recognizable.

2.5. Protocols

A survey of existing physical programming tools in the previous section indicates a lack of networking capabilities that might be used as a foundation for prototyping ubiquitous computing systems. If this is a hole that needs to be filled, let's look at a domain already rich in networked systems.

The definition of HTTP and HTML produced an explosion of networked computer applications, driven by users wielding the mighty hyperlink. Object-oriented programming created an order-of-magnitude leap in developer productivity and increased code manageability. In this vein, we need a higher-level protocol to make headway against the problems of large and dynamic networks. For ubiquitous computing to become, well, ubiquitous, there is a need for a medium to enable networked object applications, beyond the narrow interactions manufacturers produce. It should be transparent to the user as HTML is, and allow the discovery of functions and reuse of connected objects as OOP does.

The many forms of input and output that a flexible multiple-object interface enables will introduce additional complexity, and these will need to be softened in order to make them accessible. As the number of embedded computers grows beyond the number of people on this Earth, the computers are going to have to self-manage to a greater extent. [28]

2.5.1. Machine Protocols

Of the many existing services and protocols that are used for lightweight device communication, the following are representative. OSC (Open Sound Control) is a messaging format originally designed to broadcast data around a network of audio devices in real time, and has been adapted for broader use. It has a URL-like addressing scheme that features pattern matching to specify multiple recipients. Pachube is a web service and API that allows objects to publish their sensor data in a standard way on the Internet. It is intended to enable interaction between remote physical and virtual places. The Web of Things actually does use HTTP as a protocol to provide web services for objects. It suggests a standardized interface for objects accessed through an embedded

web server, with a gateway standing in for devices that don't have the resources to run an HTTP server.

DLNA (Digital Living Network Alliance) is a protocol specification that is actually in use in current consumer electronics. It is a standard adopted by much of the CE industry for using digital media across different devices—for example, playing video from a computer on a TV connected to the same network. It incorporates discovery and control of devices and services, but its flexibility is limited: DLNA is constructed around a handful of media consumption use cases, and a large industry body must approve changes.

These protocols answer some of the issues around standardization and ease of development. However, they are largely designed for point-to-point connections and optimized low-level communication—and do not tackle the challenges of how many, many more devices on the network might interact. In short, they are machine protocols that humans cope with.

2.5.2. Human-Machine Protocols

On the flip side, machines more readily adopt human protocols. Following is a selection of software agents which communicate with users through typically human communication channels.

One old example is the email list server. [29] Users can receive emails from other list members and broadcast their own. The computer that handles these functions typically has an email interface to change subscription settings. One might send an email to `list-digest@interest-group.com` that says “subscribe `user@domain.com`” in the body, and a software agent executes the desired action.

The Moviefone AIM bot is an example of a chatbot, using instant messaging to communicate with a user. One starts a chat with the bot and ask it to get summaries and showtimes for films, buy tickets or request reminders when the movie's about to start. It handles some variety in how commands are structured.

Google SMS provides many of Google's services to anyone with a mobile phone capable of texting. Send a text message containing a kind of food and your ZIP code, and you will get a list of

restaurants near you with contact information. Send a message that contains two locations separated by a “to” and you’ll get step-by-step directions from the first to the second. Send a message formatted like “translate how are you? in French” and you’ll get the desired answer. It can also solve math problems.

In fact, this works much like Google’s search function on the Web does. It parses human language and returns tailored information if it detects key words, data types and grammatical structures. It matches well-structured human phrases, so users can remember the commands easily. If it doesn’t completely understand a command, it will ask for clarifying information. The user can also send a message containing “help keyword” to get an explanation of the syntax and options.

What we’re talking about here are command line interfaces. Parsing text is trivial for computers. Given a human-recognizable vocabulary and some forgiveness in input, users have proven very capable of learning how to interact with these information systems, even when we’re supposedly in the era of the GUI. [30]

2.5.3. Machine use of Twitter

Twitter is a particular lightweight social networking service with over 125 million users at the time of writing. Its basic unit of asynchronous communication is called a tweet. Twitter provides a UI that is nearly a command line, where the user enters addressing and categorization inline with the text content of the tweet. The service is focused on broadcasting such messages with a number of innovative addressing methods to indicate the target audience. Foremost among these is “following,” which is a non-reciprocal link. A user will receive a feed of messages broadcast by all the other users they are following. Twitter’s popularity (at the time of this writing) has begun to spawn open-source alternatives which are API-compatible, [27] allowing more control over one’s own data while taking advantage of much of Twitter’s existing ecosystem.

Combined with the thorough API, the constraints of the tweet format—140 characters of text, nothing else—have unintentionally made it suitable as a machine protocol. Much of the richness of a tweet is not in the message, but in the connections it draws

between senders and receivers. (Technically, addressing is also part of the message body. Protocol transparency is another of Twitter's charms.) The following are therefore a classification of Twitter connections in the frame of the two previous sections and existing examples.

Human-to-human—Twitter was originally designed for human-to-human communication, so it's no surprise that the Twitter API has spawned countless clients for every conceivable computing platform. [31] Several consumer electronic devices now have Twitter, including televisions, cars and e-book readers.

Object-to-human—There are many software agents publishing data feeds on Twitter for humans to follow. A user can get the local weather or news, for instance. Physical objects gifted with a Twitter voice are less common. An early user-created example is a washing machine that posts a message when a load of laundry is done. [32] The Withings WiFi Body Scale posts a user's weight to a Twitter account each time it is used.

Human-to-object—Software agents that respond to Twitter messages from humans are few. One automated service that shows the potential of Twitter-based applications is TrackThis. [44] To "install" it, a user follows it, and it will follow the user back. This enables private communication via Twitter direct messages. To track a package, the user sends a direct message with the tracking number followed by a nickname to associate with it. TrackThis then sends a direct message back to the user every time the status of the package has changed. Most interesting is that the status messages generated are predictable, and therefore easily machine-parseable.

3. Exploration

The Context chapter describes current and future issues facing electronic products. I believe that these challenges can be met where product design and ubiquitous computing intersect. This chapter explores those edges with a series of experiments in meaningful interfaces between humans, physical objects and data. These projects illustrate the role that designers are fit to play in the creation of future computers.

3.1. Presocial objects

Through these early experiments, I uncovered threads which organized my thinking for how to build subsequent projects. Eventually, I spooled enough to stitch together the principles collected in Synthesis.

3.1.1. Kinesic Interface

We tolerate technology as much as we embrace it. Every new device is intended to make our lives easier and more enjoyable. Yet interacting with our devices in public often communicates separation to the people around us.

The Kinesic Interface is an exploration of gestural interfaces that reintroduce a civility to human-machine interactions. It challenges the barrier that the operation of consumer electronics often puts between humans, and proposes overlaying human-object interactions onto human-to-human interactions using body language. This is a design concept that I did in collaboration with Abe Camacho. It's part critical design, part speculation on how humans might interact with devices in a way that's human-readable as well.

Camera

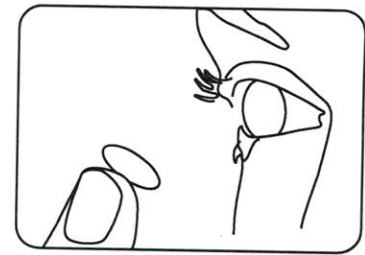
When you use a typical camera, you place it between yourself and your family and friends—removing yourself from the picture, so to speak. But integrate the camera into eyeglasses or contact lenses, and you remove the divide as well as the inconvenience. The popularity of cameraphones has shown that people value availability—having a memory preserved with little effort is much more valuable than a 5X zoom or 8 megapixels.

Take a picture by winking at your subject. Have a stranger take a picture of you and your spouse by telling him to wink at your wife. [Figure 14]

Media player

Fiddling with controls is distracting, and not an ideal situation while driving or going for a run. With a gesture-based interface read by a motion-tracking ring on your finger, you can keep your eyes on the road and your balance on the treadmill.

Adjust the volume by sticking your finger in your ear and twisting it, as an old man might do when he can't hear well. Pause and resume your music by plugging and unplugging your ears, or by tugging on your earlobe. Advance through songs or stations by twirling your forefingers, much as you would if you were impatiently telling someone to “get on with it.” [Figure 15]



Apply contact lens interface



Wink, take photo

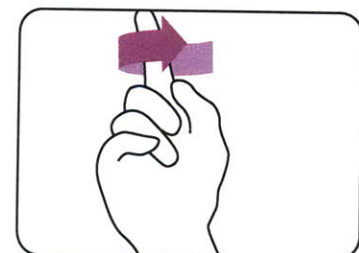
Figure 14. Camera interaction.



Volume Up & Down (Rotate Finger)



Pause & Play (Pull earlobe)



Forward & Back (Twirl Finger in Air)

Figure 15. Media player interaction.

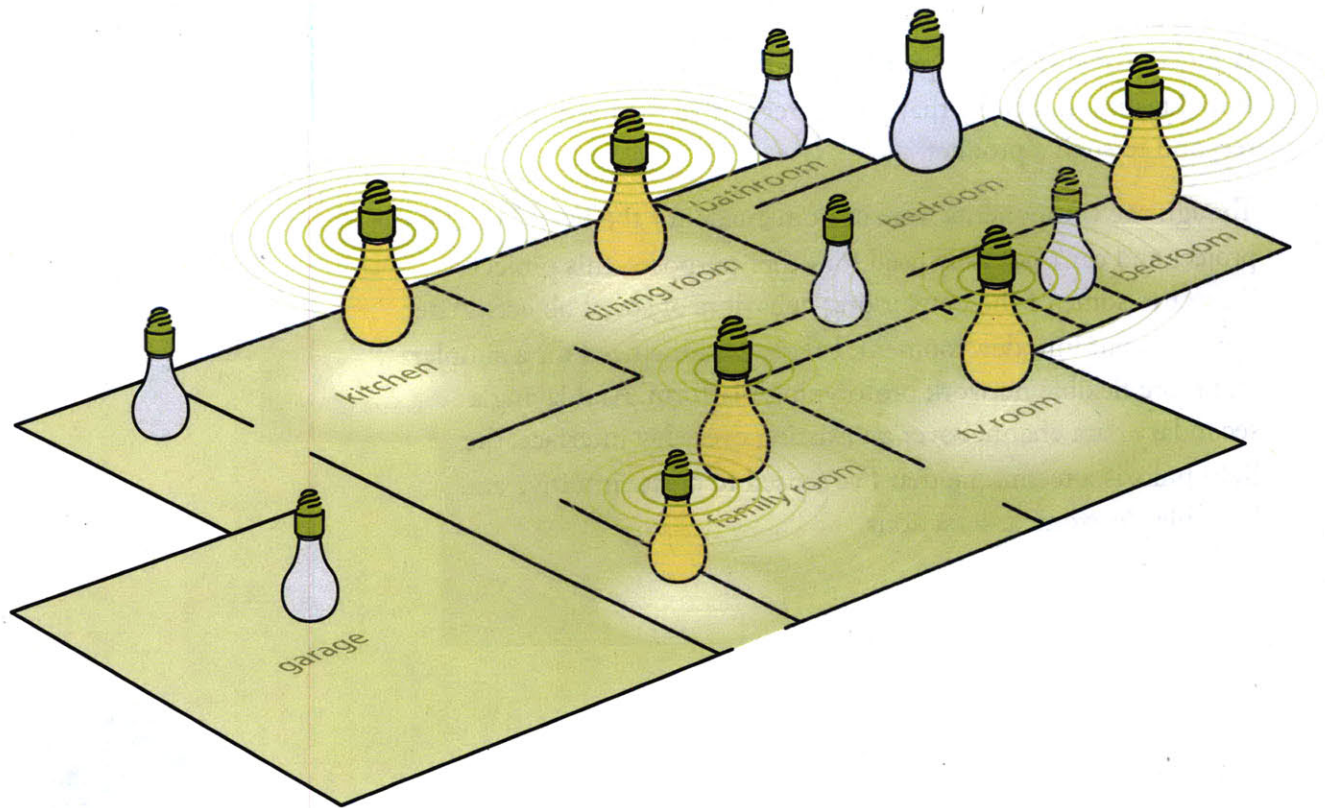


Figure 16. Watt Watchers system diagram

3.1.2. Watt Watchers

People are aware of the need to reduce their energy use, but a wealth of consumption data does not effect a change. And devices that automatically shut off without the user having to make a decision address the symptom, not the problem. How can a user be made conscious of his consumption habits in order to change his overall behavior on a deep, actionable level?

Watt Watchers is a network of light bulb collars [Figure 17] that collectively dim as more lights are turned on, to encourage better consumption habits. Using lights as a proxy for household energy use, the system reminds users when they are consuming above average power by mimicking a brownout. The user's input into the system is the light switch, and when the lighting threshold is too low, he or she can wiggle a switch to raise it. [Figure 16]

Using a Zigbee wireless mesh network, the system observes and adjusts to the user's behavior to avoid getting tuned out. It is decentralized: the lights' only communication is to broadcast their state, which each lightbulb uses to make its own decisions.

This is an earlier work done with Jordan Fischer and Sarah Jones. A functional system was prototyped and installed in users' homes. Initial testing indicated that people recognize the need for and respond to such a product.

Though the components were custom-made for the Watt Watchers project and can't be easily used for other purposes, this project illustrates some of the networking behaviors of social objects. After the effort put into development, I registered the need for a simpler and more flexible network prototyping platform. And laying a secondary data channel over an existing everyday interface, the light bulb, is a technique that I've reused to create intuitive and invisible electronic interactions.

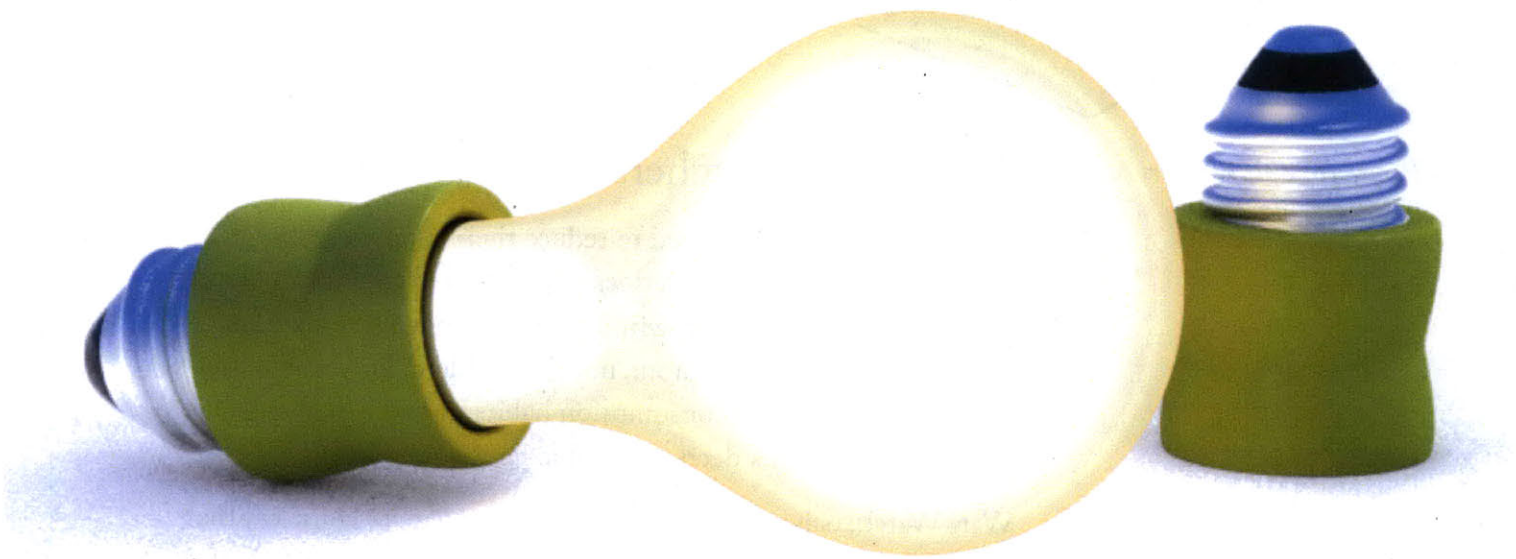


Figure 17. Collar concept with wireless node.



Figure 18. Social Garden client for a personal computer.

3.1.3. Social Gardening

The Internet supports many great tools for communicating at a distance in order to maintain personal relationships and build social networks. However, these tools rarely help us realize which relationships are strained by lack of attention. Staying in touch has become a chore.

Social Gardening explores using virtual plants as a metaphor for relationships, encouraging users to cultivate social connections as we do our gardens. It is software that tracks and analyzes communications through email, instant messaging, social networking sites, SMS and phone, in order to give feedback on the health and nature of relationships. It condenses the large volume of communication that we amass over time into a simple, warm visual display.

The Social Garden application aggregates personal interactions across many channels through a computer or smartphone. For each relationship, it creates a meaningful visualization that provides feedback on how the relationship has changed over time, encouraging the user to maintain their garden. It compares the frequency and content volume of recent communication with the average of all communications with this person, and so when these deviate from the average, the associated plant will reflect that by growing or wilting. The visualizations can be delivered as ambient information on a consumer electronics device.

Social Gardening comes in several flavors. A Macintosh application plays the center role of processing the data about the communications in addition to drawing the visualization at the bottom of the screen. [Figure 18] The user selects contacts from the Mac OS built-in address book, and the application uses those

friends' email addresses, IM handles, and so on to record the metadata in communications between the two in the background. Clients for Symbian Series 60 phones and the iPhone provide call and SMS logs which are synced to the Mac client for processing. In a vertical orientation, the iPhone version allows the user to browse their garden of relationships one plant at a time while. In a horizontal orientation, the garden shows all relationships rendered in a treemap, [66] resembling an aerial view of plots of land. The size and greenness of these plots are determined by historical and recent frequency of communication, respectively. [Figure 19]



Figure 19. Social Garden client for a smartphone.

And for demonstration/exhibition purposes, a real-time version was made in the form of a Java chat client. [Figure 20] Two people on different computers can have a conversation through the chat client, and a plant on the screen grows in response to the volume and timing of the conversation. The local user provides the input of water, and the remote user's input is tied to sun. If the conversation lags, the leaves gradually fall off. If it becomes too one-sided, the plant wilts or yellows. And if the conversation takes a negative turn, thorns appear on the plant. This last feature draws on the Twitter Weather web service which gauges the emotional content of what's typed. This can optionally be fed an XML file instead, to generate visualizations from any three variables.



This project attempts to address information overload by smartly reducing data from many sources into a few relevant bits. These are rendered in a rich visualization that uses a familiar design language. Several variables are contained within an image of a plant, but we intuitively understand what a healthy or neglected plant looks like.



In trying to provide an experience that fits in the flow of life, Social Gardening encountered another challenge familiar to ubiquitous computing systems: the effort of writing a single application across many different devices. Clients have been written for multiple devices in order to collect a more complete picture of a person's communication, and to give feedback whenever it might be appropriate. This experience influenced the architecture of my other network-enabled projects, particularly separating the data processing service from the input and output nodes, and standardizing the communication between them.

Figure 20. Social Garden real-time chat client.

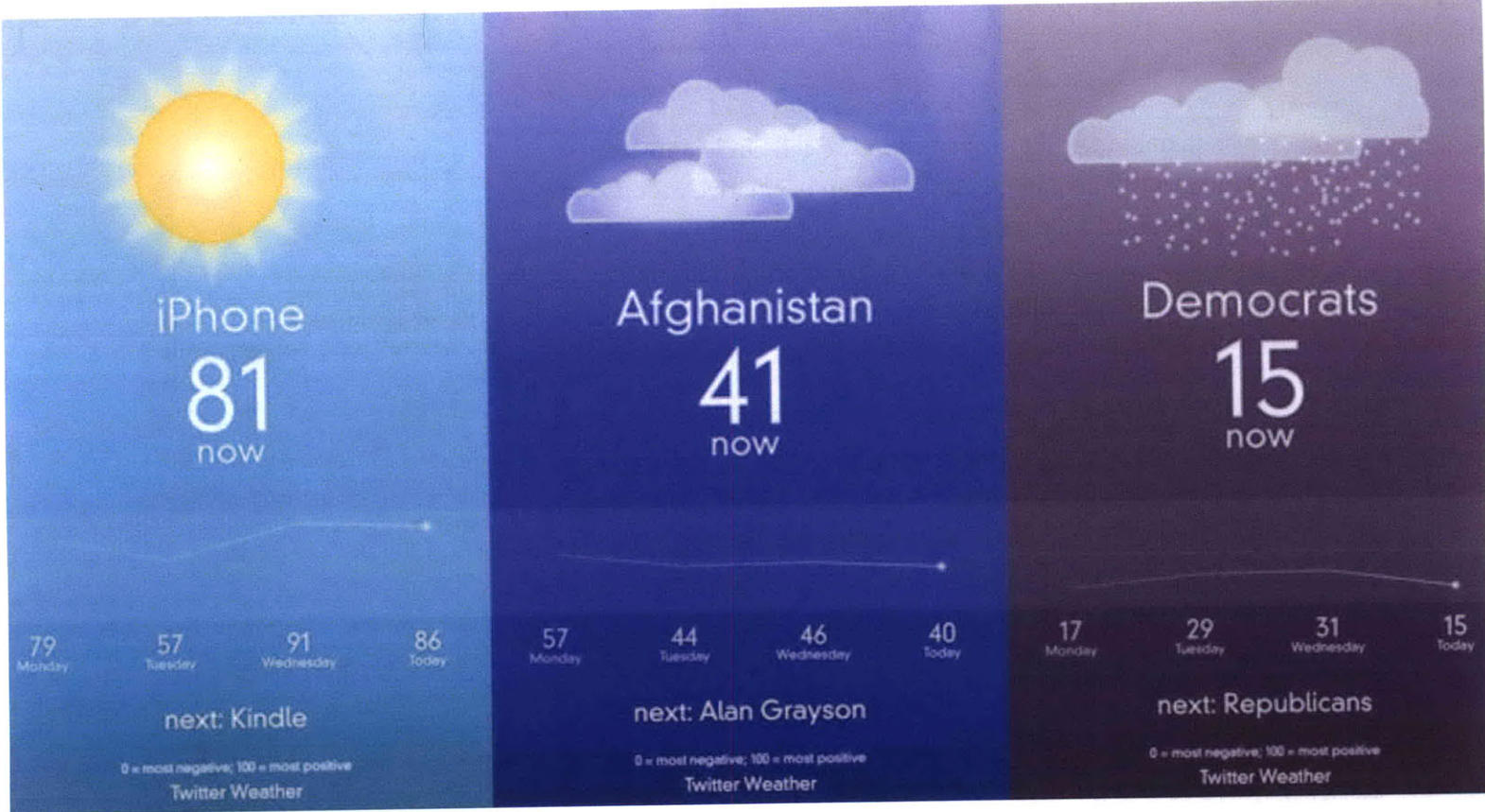


Figure 21. Screenshots from Twitter Weather visualization for kiosk.

3.1.4. Twitter Weather

The vast amounts of user-generated content on the Web often produce information overload as frequently as they provide enlightenment. Twitter Weather reduces large quantities of text into the prevailing mood of the content, and visualizes it by rendering a weather-report-style display.

The primary experiment is a dynamic Quartz Composer visualization [Figure 21] and a website [Figure 22] that, when given a topic, returns the emotional valence associated with that topic on Twitter at the current time as well as the last three days. This data is rendered as a “weather report.” Positive and negative feelings about a topic are mapped to a “temperature” from 0 to 100.

Comment Weather is a bookmarklet that gauges the emotional weather of the comments on an articles for some comment-supporting websites, such as the Wall Street Journal and Engadget. When reading an article, the user clicks on the bookmarklet, which scrapes the comments from the webpage or RSS feed, processes them, and inserts a weather report into the page. [Figure 23]

Twitter Geowater is a series of visualizations of Twitter weather on a topic, mapped by location of tweets within the U.S. The visualizations use Twitter’s location filter to retrieve and process



Figure 22. Twitter Weather public website.

only tweets made within a given radius. Each circle approximately represents a 400km-diameter area. Green is positive mood, and red is negative mood. These are snapshots of a manually run Quartz Composer visualization; the amount of data required for on-demand visualizations quickly exceeds Twitter's allowance for a user. [Figure 24]

The Twitter Weather projects are supplied with raw scores by an internal web service which uses a Bayesian classifier to aggregate and rank emotional content on a topic. The classifier is trained through a webpage which allows users to rate random tweets. It was initially seeded with several thousand ratings by students. The ANEW affective wordlist [67] was initially used, but its 2000-word dictionary proved to be limited and inflexible. The Bayesian classifier, written by Stephanie Bian, currently has about 10,000 entries. With users regularly rating Twitter content, it can keep up with the dynamic vocabulary, spelling and meaning of living languages. This is especially important on Twitter, where a casual environment and length limitations often produce messages unintelligible to a proper dictionary.

While pinpoint accuracy is not my primary focus, Twitter Weather does a reasonably good job with enough source material. There are several ways to improve results, but using context to calibrate the scale would be useful. Internet comments tend to be negative, so users are told to think of emotional weather as the climate in London in March—50 degrees is a great day.

Like Social Gardening, this project tries to reduce a complex dataset to something understandable at a glance. Quantifying the mood of a large group of people is particularly useful for making intelligent decisions in software agents. Using it to calculate thorn growth in Social Gardening motivated further exploration into exposing the useful algorithms of other standalone products as services. And I got a taste of the power in the self-organizing behavior of Twitter users through simple messages.



Figure 23. Comment Weather on a Wall Street Journal online article.

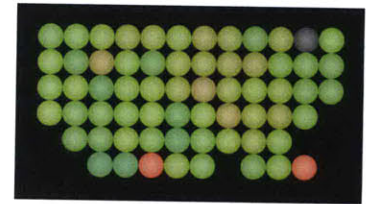


Figure 24. Twitter Geowather for Scott Brown, a candidate for Massachusetts senator, after he won office.

3.2. Tools for designers

Product designers are now service designers. We can't help it. Most of the devices we're making now are part of a network, or need to work with other devices, or at a minimum have a website. Our products are now services.

—Dan Saffer [1]

In the introduction, I talked about democratizing control of our electronic objects, lowering the barrier to programming them at a high level. I believe this starts with product designers. As the craftsmen of the human-object interface, they should play a guiding role in the construction of electronic objects centered around human functional and emotional needs. [18] Product designers have the skills and perspective to lead the discussion on what these new forms of interactions can look like, but do they have the tools?

Through the research described in the “Prototyping electronic objects” section and my own experiments covered in this chapter, I conclude that they do not. This section describes tools that I have developed in response. My goals here: Make talking objects easy. Make object conversations possible.

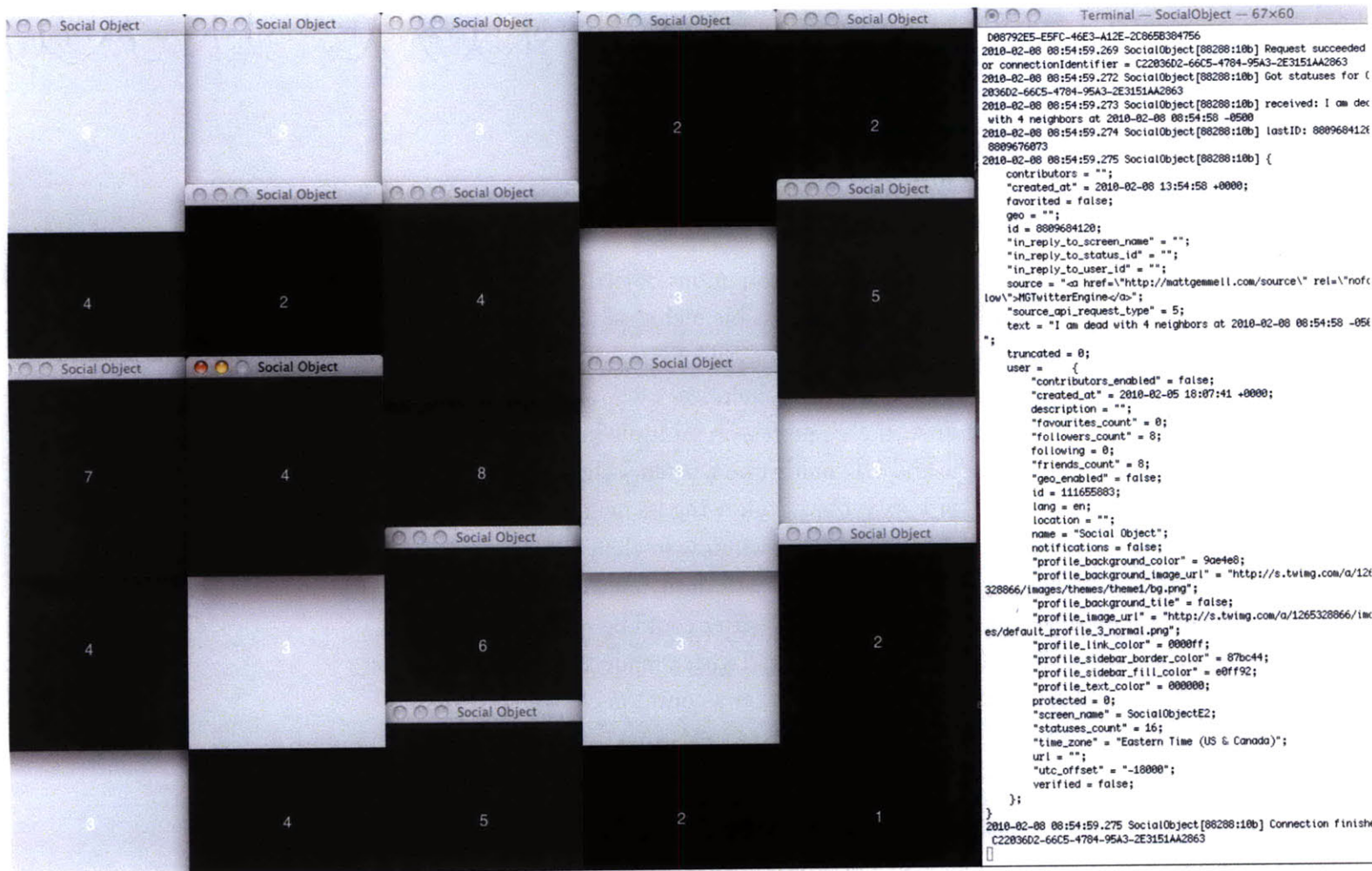
3.2.1. Hand/Fingers

In this thesis, I describe a practical vision of ubiquitous computing with tangible interfaces. One configuration of this orbits around an individual, mediated by his or her personal consumer electronic devices. I developed the Hand/Fingers platform to aid in designing devices for this model. [84] This toolkit simplifies the process of networking objects, both at the developer and user levels. It eliminates the effort that programmers duplicate in inventing a new communications protocol, and makes it possible for users to create their own connections between objects to achieve tasks that the developers did not anticipate.

Finger is a communications library for the Arduino embedded computing platform [16] which handles remote discovery and execution of local functions. A programmer includes the library, and calls a setup function once for each Arduino function to be made public. After attaching a Bluetooth module or USB cable to the device, the functions can be called remotely, and a blank message returns a list of commands it understands. Finger handles all serial communications, but if the programmer wants to also process serial data, she can provide a callback function. I'll refer to objects created with this library generally as Fingers.

Hand is routing software written for Processing/Java clients and Symbian Series 60 phones to relay Finger calls and their results through Twitter. It also serves as a framework to write software agents that reside on Twitter. The software uses open-source Twitter frameworks Twitter4J for Java, and Python Twitter for Symbian. For each attached Finger device, a programmer creates instances of the Finger class, passing the serial port that the device is attached to. This handles connections with the Finger and Twitter. Most importantly, Hand passes messages that appear in the associated Twitter account's stream (from other people or objects that the device is following) down to the Finger, and posts messages from the Finger to its Twitter account. The programmer can write logic to process the messages before routing them, or not route them at all.

Several of the experiments described in this chapter are social objects. These were built with the Hand/Fingers toolkit, and the toolkit was improved through each experiment.



3.2.2. Virtual Social Objects

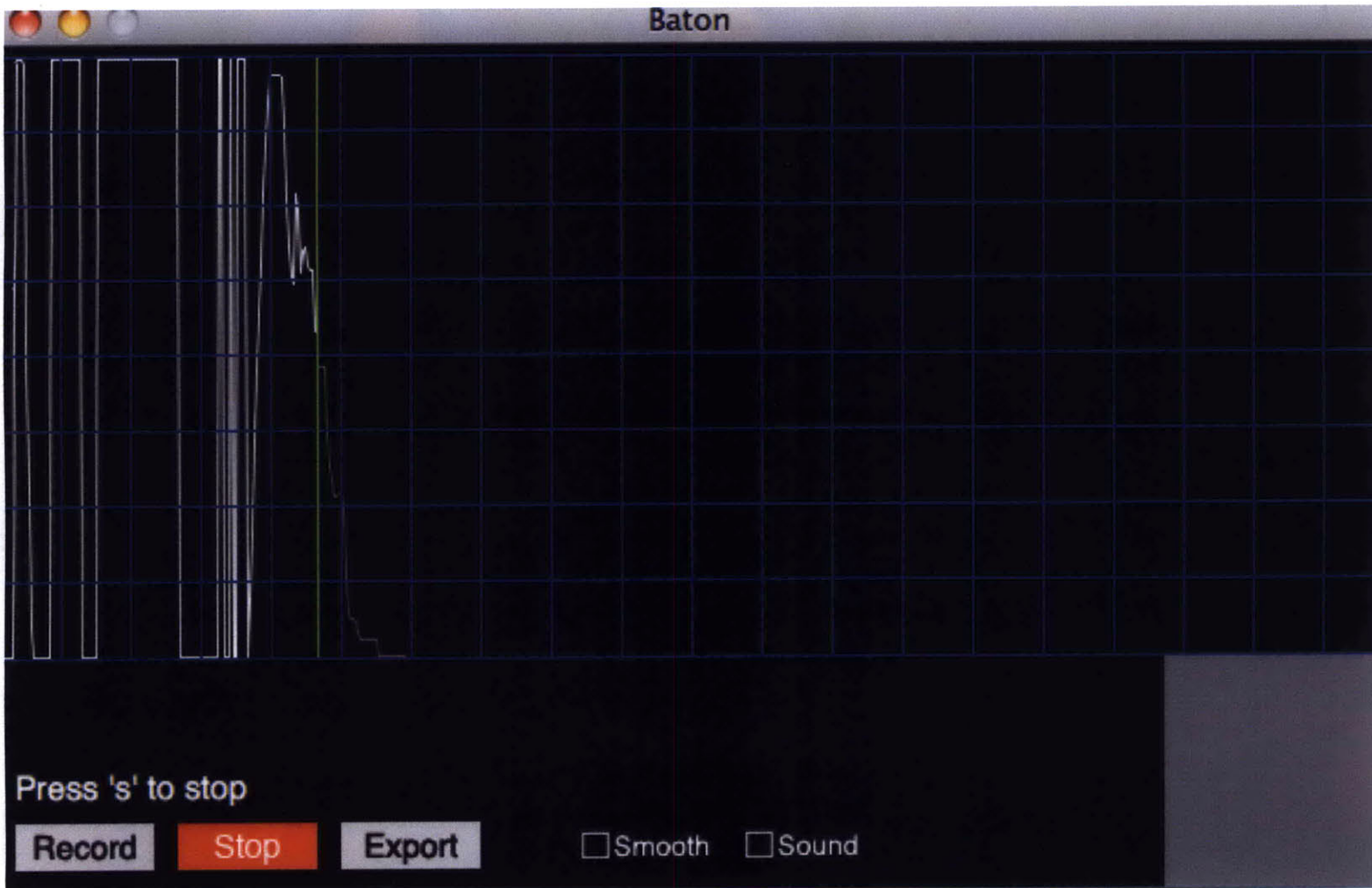
This is a Macintosh framework written in Objective-C that serves as a software testbed for networked objects. It is a Twitter client with a minimal but customizable user interface, providing a foundation on which the developer writes logic and virtual inputs and outputs to simulate a Twitter-connected device. It can also be used to build a Twitter software agent.

When launched, it logs into the Twitter account provided for the object, and regularly processes new messages. By default, it logs incoming and outgoing Twitter activity in a text window. It leverages the human-accessibility of Twitter, providing a “terminal” for the developer to post messages on the object’s Twitter account for direct control. The framework also provides posting functions for the logic code to call. The developer may use a graphical window to reflect the simulated object’s interface.

The Virtual Social Object framework allows a designer to quickly test interactions within object ecosystems. The user can create a custom UI in Apple’s Quartz Composer, a visual programming tool

for interaction designers. [45] It might present a model of a physical object, or a purely digital one. Writing rules requires a knowledge of string manipulation using Objective-C.

I initially built this as a petri dish for autonomous object ecosystems in which I could pinpoint possible technical challenges. As a proof of concept, I constructed a Conway's Game of Life simulation from 25 virtual social objects, identical except for their Twitter accounts. (Of course, there's no reason multiple kinds of objects with custom logic and UI couldn't be written.) These were arranged in a 5x5 grid, with each object following its neighbors on Twitter. These made use of the graphical window to change color depending on whether the object was alive or dead, and the simulation ran as fast as the feeds from Twitter could be refreshed. The latency of the network, mostly due to Twitter, made the objects update out of sync. This indicates that either a common clock is needed, or this model should not be used for real-time applications.



3.2.3. Baton

In exploring expressive physical output with several of the projects that follow, I grew frustrated with the clumsy process of translating ideas into code, uploading to a microprocessor to test the quality of the translation, tweaking and repeating. This was unnecessary friction in the iteration loop which is a staple of design. My observations of the interaction prototyping process, discussed in the “Prototyping electronic objects” section, provided an insight: Our fine motor control makes hands extremely expressive, so these are natural instruments for authoring digital interactions.

Baton is a sampling software tool I wrote to help designers use their hands to create expressive digital output. (No pun intended.) It maps manual input such as tapping on a key, moving the mouse, or waving a hand to a digital output. This may be a pin on a microcontroller that operates a motor or an on-screen character opening and closing its mouth.

Because the purpose of this project is to lower the barrier to developing rich physical electronics, the primary output device

targeted is the Arduino microcontroller. Baton can send it samples over a serial connection in real time for direct feedback, and export an array of values along with sample playback code on an Arduino board. The software also provides visual and audio feedback while recording or playing back samples. It is written in Processing/Java and for input, uses a computer's keyboard and pointing device as well as the ambient light sensor in the MacBook Pro's speaker grille (which can roughly detect the height of a hand over it)

Baton was used to make the tangible output of several of my projects, including the Proverbial Wallets and the Connected Dashboard.

3.3. Social objects

Earlier work defined threads of composite human/machine interfaces, condensing data, and networked interfaces. These experiments follow those threads deeper, each illustrating how they come together into social objects.



3.3.1. AsDrawnOnTV

People often use shared television experiences as a backdrop for conversations. But with the increasing popularity of social networking technology, television is competing with a diverging field of social media. Television is broadcast at the viewer, and interactivity has been limited to how you can watch. It's remained one-way communication. how can the viewer talk back, and create new shared experiences?

AsDrawnOnTV is an application for television that uses video content as the shared canvas on which people converse. The prototype built uses a telestrator-like doodling interface, but several types of real-time input are possible. The user's input is recorded along with the underlying video. The user can comment on anything she is watching, and share it with friends along with the underlying video, using Twitter as the transport. When shared, these videos with commentary pop up as an additional channel on your friend's television, available for immediate viewing or reply. [Figure 25] This is intended to make sharing and annotating native to the device on which most people consume video content.

The technology behind this uses a set-top box application and a remote with accelerometers (prototyped with a Wiimote and Processing) to draw and control the interface with simple gestures. Videos are shared through Twitter, which allows friends to see your video clips on any connected media device. When the user ends a

Figure 25. Drawing on the screen and receiving a shared video via Twitter.

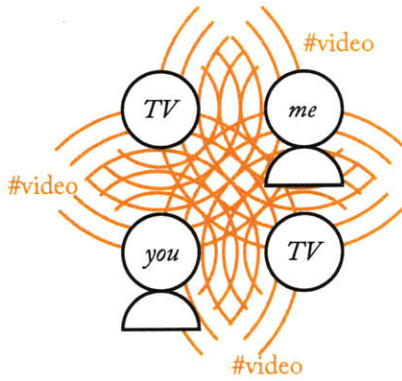


Figure 26. Social system diagram. Video URLs are shared with all followers, with a tag to identify the content.

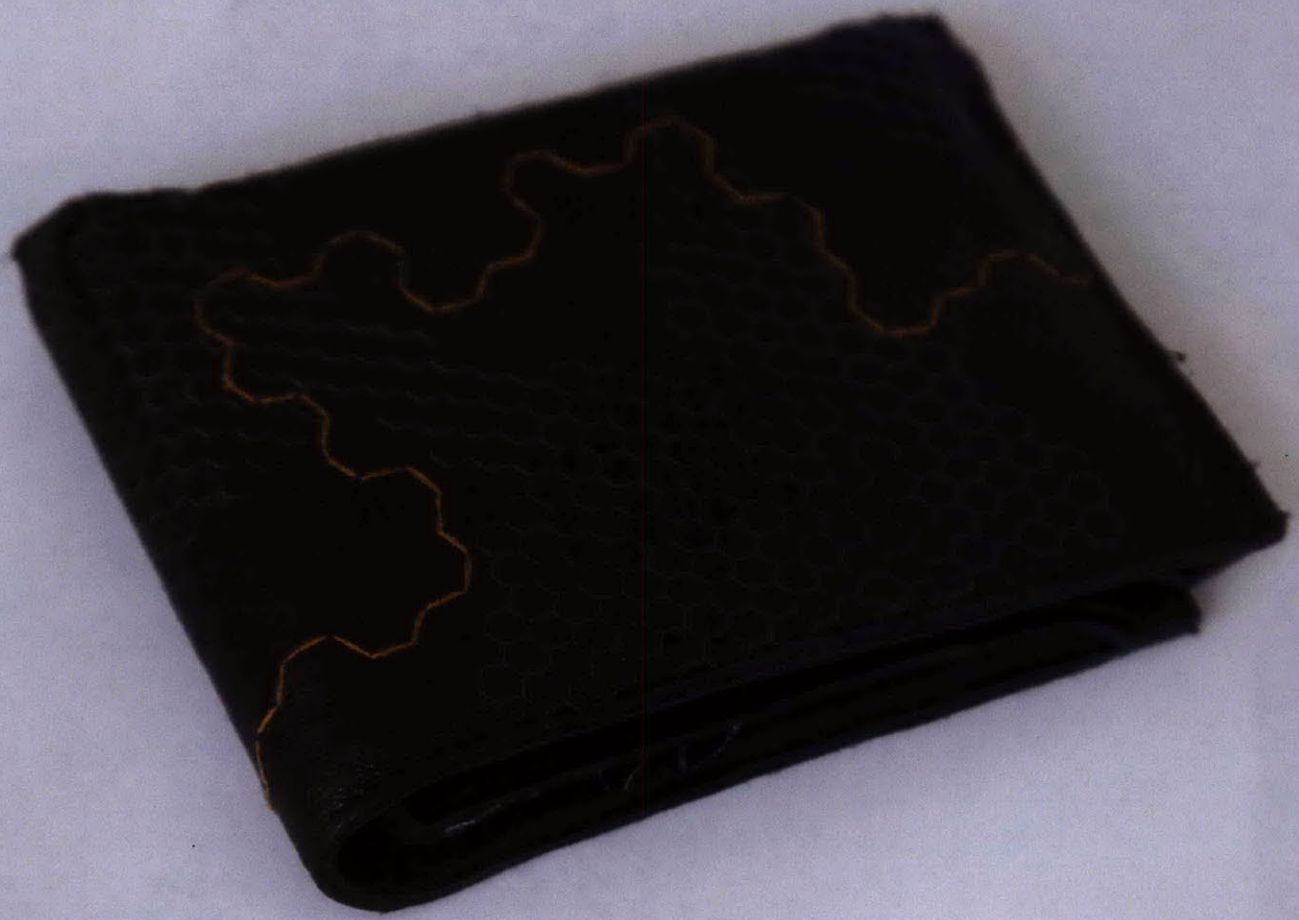
doodle, the video is uploaded to a server, and the application posts a link to it on the owner's Twitter account. This allows another user's copy of the application to read and display the videos posted to friends' Twitter accounts directly on his or her television, or if the user does not have the application, he or she may view it directly via Twitter. [Figure 26]

This builds on the Kinesic Interface's concept of a language understandable by both humans and devices, but using a computer-native medium. This was my initial experiment with using Twitter as a common transport for people and devices, so AsDrawnOnTV is the first social object I developed. It predates the Hand/Fingers framework's use of keywords to call functions, but if updated might use a #video hashtag to make recognition of video URLs easier.

- *Input:* adds any video URL to a playlist.
- *Output:* broadcasts a URL of a video clip + doodle generated by the user.



*Figure 27. Mother Bear
Proverbial Wallet.*



*Figure 28. Bumblebee
Proverbial Wallet.*



*Figure 29. Peacock
Proverbial Wallet.*



Figure 30. Mother Bear Proverbial Wallet, open.

3.3.2. Proverbial Wallets: Tangible interface for financial awareness

There is often a disconnection between financial decisions and consequences in our consumer culture. We used to know how much money we had because we physically carried it in a purse with us. Now, many of us are not diligent about balancing our checkbook or watching our account balances. With credit cards and online banking, our idea of money has become abstract, and the transaction experience does not reflect the amount of money involved. [50, 48] When we're shopping, do we know what our bank account balance is, or whether we're over budget for the month? Our existing senses are inadequate to warn us.

The Proverbial Wallets reintroduce physicality to virtual financial assets. The wallets connect personal financial information to tactile feedback, such as incoming and outgoing transactions mapped to vibrations, or a monthly budget mapped to rotational resistance in the wallet's hinge. This is intended to develop a subconscious touch

memory that guides responsible decisions. In addition to providing a visceral connection to virtualized money, tactile output literally keeps personal information in your hands, private.

I wanted to imbue dumb objects with intelligence, rather than create a consumer electronics device. B.J. Fogg describes the principle of *kairos* as “finding the opportune moment to present your message.” [46] The Proverbial Wallets use this principle in an attempt to guide the user toward responsible decisions, by delivering the appropriate message at the appropriate time.

The wallets started as a collaboration with Daniel Leithinger, and as I carried it further, Danny Bankman and Emily Tow contributed their electronics and fabrication talents, respectively.

A material human-computer interaction

Haptic actuators are used to connect intangible virtual transactions to the user’s physical world. At the same time they provide a private channel of communication for sensitive data. The Proverbial Wallets do not prevent the user from spending money, but instead provide subtle ambient information in an appropriate context.

Much as Dunne and Raby’s *Compass Table* and *Nipple Chair* bring ignored electromagnetic fields to life through objects that provoke emotional responses [49], the wallets use feedback that gives character to financial information. The struggling and twitching of the wallets might produce a similar user response.

This project explores what meaningful information haptic feedback can transmit, given its limited resolution. The wallets answer this challenge in two ways. First, they provide a context that indicates the nature of the information. Vibration in a cell phone could be an incoming call, or some other notification. In a wallet, it’s clearer that this relates to financial data. Second, the information chosen is ruthlessly reduced to a single axis per actuator, such as cash flow or monthly budget. This lack of modality further eliminates ambiguity about what is being communicated.

How it works

Each of the Proverbial Wallets contains a microcontroller, rechargeable battery, Bluetooth module and haptic actuator. Tactile feedback provides ambient awareness of the user’s account balances

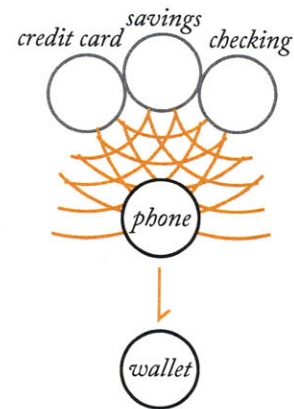


Figure 30. System diagram. Bank agents with protected accounts, that only the owner can follow, post events for the phone to pass to the wallet.

and transactions. The electronics are compact and tough enough to sit on. The Hand/Fingers toolkit's first iteration was developed for the wallets. These use the Finger library to expose commands to trigger varying tactile feedback.

Unlike most Bluetooth devices, the wallets are not peripherals for phone functions. Software using the Hand framework runs on the user's smartphone in the background, retrieving bank account information from the phone's Internet connection. The phone then calls the exposed functions via Bluetooth, which reflect the information in haptic feedback. The two devices coordinate to provide a seamless service to the user.

3.3.2.1. Mother Bear

Mother Bear protects her offspring, and her stores for the winter, in a tight grip. [Figure 27, 30]



Figure 31. Mother Bear interaction.

This wallet protects the money within it, making it slightly more difficult to open when you need to be thrifty. The resistance increases as a given budget threshold is approached. This behavior promotes ambient awareness of how the user is meeting her monthly goals as she's opening the wallet to spend—more timely feedback than the periodic bank statement. [Figure 31]



Figure 32. Mother Bear hinge with gearbox and shorted motor.

The resistance feedback is generated through a passive circuit with a small gearbox and a DC motor acting as a hinge. [Figure 32] When the motor is shorted, the hinge offers more rotational resistance, making the wallet harder to open. The motor does not require power, as opening the wallet generates a counter-electromotive force in the shorted motor to opposes the motion. A future model might eliminate the need for external power for the wireless connection and microprocessor by harvesting energy from the opening and closing of the wallet.

- *Input:* keywords *strong* and *weak*, which adjust the hinge resistance.

3.3.2.2. Bumblebee

This wallet buzzes whenever the bank processes a transaction for the user's account. [Figure 28] This creates a tangible connection between a user's action and the financial consequences. [Figure 33] Two different vibration patterns represent debits or credits, and the

intensity correlates to the amount of the transaction—a long buzz may mark a paycheck, and a shorter one may represent a lunch. If the buzzes don't match the user's actions, it may indicate that his spouse or child is using the credit card, or that fraud is taking place.

The vibration feedback is generated by an eccentric motor. The intensity is less than that of a cell phone, since it is ambient information and not a request for action. Determining the pulsing patterns took some experimentation, made easier by the Baton software. They were first modeled on the Doppler effect—money leaving the user's account would trigger increasingly longer pulses, and money received would trigger increasingly shorter pulses. This was not clear to users. Since they would have to remember a mapping regardless, I chose two mnemonic couplets to represent deposits and withdrawals. The motor is not very good at playing notes, but does provides a distinctive rhythm.

- *Input:* keywords *buzz* and *buzz <value>*. The former produces two short bursts of the motor, used for general alerts. The latter takes a number. Positive numbers play the “Charge!” rhythm and negative numbers play the “Shave and a Haircut” rhythm.

3.3.2.3. Peacock

The Peacock displays its assets on its backside, to attract potential mates or intimidate others. [Figure 29]

This wallet swells and shrinks to reflect the user's account balance. The effect is that the user will feel a subtle tightness or looseness in his pocket that persists until the related account balance changes. This promotes a sense of general financial health and may encourage users to save money. [Figure 34]

The feedback is generated by an embedded servo, which rotates an arm to adjust the gap between two plates. Nine addressable positions cover the range of thin to fat. Because of the size of the mechanism, this wallet is more of a critical design piece than the other two, reminding us of less practical, more emotional aspects of using a physical wallet.

- *Input:* keywords *grow*, *shrink*, and *size <value>*. The first two increment the position of the servo, and the last one moves the servo arm to a position, mapping a number from 1 to 9 to 0 to 90 degrees.

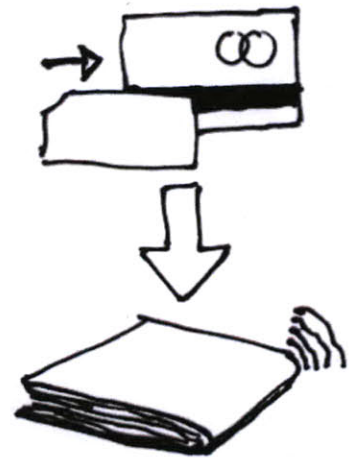


Figure 33. Bumblebee interaction.

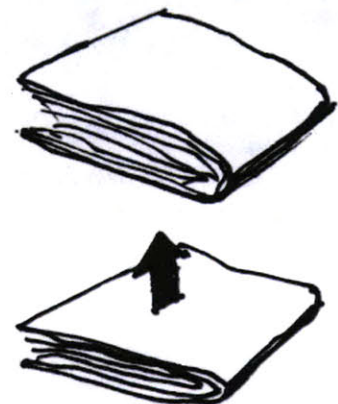


Figure 34. Peacock interaction.



Figure 35. Proximeter social awareness instrument.

3.3.3. Proximeter

Social networking services such as Twitter and Facebook provide a constant, low-resolution stream of what friends are doing. While this would seem to induce information overload, these services have developed in users a social proprioception [53], a sense—not a thought—of the general state of our tribe that we have struggled to retain in globalization.

The proximeter is an instrument in the mold of the old astronomical tools described in “Craftsmanship of electronic objects,” giving the owner a sense of the movement of his or her social cloud. It is a software agent that tracks the past and future proximity of one’s social group through data streams such as shared calendars. And it is a physical instrument that charts this in an ambient display, drawing a glanceable pattern of paths that tells the user how near their friends and loved ones will be in the near future. [Figure 35] The materials are an important part of the

experience, as is hiding the complexity of the computer which enables social interaction.

Interface

The proximeter has a square display, in which the y-axis represents time and the x-axis represents distance. A vertical rule down the middle represents the Here, where the owner is. Filaments representing individuals in the owner's life run vertically, weaving back and forth as they trace the proximity of a person to the user in the x-axis. These are similar to sparklines [52], which compress the movement of a data point into a less-detailed graphical representation.

Proximity can be used to describe relative distance in space, or the amount of time it might take one to cover that distance. I can describe the proximity between myself and my mother as 2900 miles, but a more practical description might be that we are 6 hours apart, or even a plane flight apart. Time progresses in the y-axis on the display, with a tuned logarithmic scale used to group lines by convenience of access, such as those people that are a walk away, or a drive away, or a flight away. A horizontal rule about three-fourths of the way down represents the Now, leaving room to see three weeks forward and one week back.

Visualization

The proximeter can be read at several levels of attention. At a glance the user can see whether the rhythms of friends remain steady by the filaments running parallel. An intended change in location will register as a bend in a line. By default, all filaments are shown, without any text. A knob on the frame is used to browse left-right through filaments, displaying the name and current location of the associated person or object. Clicking the knob switches to browsing up-down through a filament's timeline, stopping at each change in location or event and displaying a description. [Figure 36] Because the proximeter is primarily an ambient display, this is a reasonable amount of information with which to make a decision. Recent proximity updates brighten the filaments of the associated person, while static data feeds fade.

The instrument's functionality stops here. It does not attempt to enable communication between the user and the selected person.

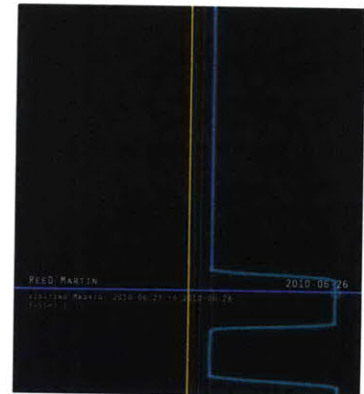


Figure 36. Browsing a friend's proximity history.

This is intended to avoid becoming an agent that exposes every bit of information about a person. This information may be willingly published, but without a personal exchange, it can feel intrusive. The abstraction of data displayed by the proximeter provide a manageable level of privacy and user discretion. While geolocation check-in services such as Foursquare [64] may be used as data sources, the proximeter is designed to provide a more ambient experience, and one that projects where people will be in the future.

Scenario

As Gabe enjoys his morning espresso, he glances at the proximeter hanging in the foyer. As usual, most of its filaments are all roughly parallel to Here. Some filaments are bunched near the center, representing local friends in Seattle. A few are further out, representing people in other metropolitan areas or states. He can see one filament on the left edge that he knows represents Kayo, since she's the only friend that far away.

Gabe then notices that one bright filament swoops in from the edge just above Now (crossing even Kayo's line) runs parallel very close to Here for a while and then dips back out a bit and settles into a parallel line slightly further away. From this he can surmise that a friend is soon flying into Seattle from beyond Japan, then soon after flying out again to somewhere else in the States. Seeing an opportunity to reconnect with someone, he walks up to the proximeter and turns the knob until the line of interest is highlighted. Jordan's name comes up, and Gabe might send an email to make plans to meet up while he's in town.

Materiality

The proximeter experiments with reframing consumer electronics as companionable instruments, through materials, interaction, and focus on a narrow human domain. Here, the domain is the presence of loved ones. It channels a bit of the warmth that radiates from the motion of people through our lives, when they're far away. Inspired by wonderful old scientific instruments in the Whipple Museum, [39] it uses pre-industrial materials and has a specific function which it will retain for a long time through silent software updates and a narrow scope. While it contains a computer, the display is built out of brass and glass. Carrying this aesthetic to the

software interface, the glass and a vellum layer are used to disguise the flat pixelated nature of the LCD screen beneath. The visualization tracing friends' paths through time and space are rendered as bright, trembling filaments.

The proximeter draws on a different design vocabulary from that of the iPod Touch described in the "Materials in electronic objects" section. It's shiny, but it's not aloof. The materials commemorate where it's been touched. The object will take on a patina that says it's good enough to have been used a lot. (If anything, I wish it could show more wear. A CRT display would get burn-in at the distances that people tend to be from me. The detents in a mechanical knob would get worn down at frequently checked friends, and the knob would more readily fall into those grooves.) The feel of materials beneath our fingers, and our perception of other physical attributes, allude to objects and experiences in our memory banks. They may not inspire awe or lust. But that's the point. They're just tools.

How it works

The object is built around a tablet PC running an application written in Processing which fetches data over the Internet and renders it to the display. People and objects to follow are assigned by pointing to location-based feeds through a built-in Web server. To track people, the proximeter uses the Dopplr travel logging site, where users share their trips with a social circle. Dopplr's API will provide a list of all the user's friends and travels, future and past.

The proximeter also has a Twitter account, which provides a simple way to add other data sources. From any messages it receives, it tries to parse out a name (which does not change for subsequent location updates), date, and location which is converted from text to geographic coordinates using the Yahoo! PlaceFinder service. It currently is connected to the TrackThis web service's Twitter agent, to track the movement of packages that have been shipped to the owner. [Figure 37] As the proximeter's two axes are space and time, it would be well suited to tracking spimes using data sources such as SourceMap. [63] The Twitter interface makes it a social object in terms of output, at least.

- *Input:* Dopplr feed via API calls, Twitter direct messages with a name, date and location.

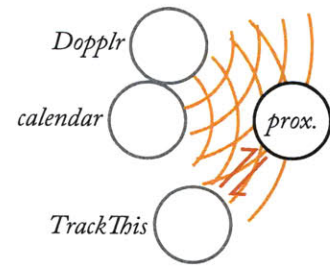


Figure 37. System diagram. The proximeter follows software agents tweeting travel updates for people or packages.



*Figure 38. DoingDoing 7-bell
alarm clock from hell.*

3.3.4. Daydar: framework for socially motivated productivity

Attempting to meet goals is a constant source of stress in people's lives. One contributing factor is the increasing isolation in which tasks get done. The proliferation of email and telecommuting means that there are less of the informal and indirect interactions—from overhearing a negotiation on the phone to feeling the stress levels around the office—that provide useful learning experiences, rhythm and feedback to work. [68]

Daydar is a critical design exploration done in collaboration with Richard The. It involves a variety of physical and digital artifacts which act as input and output for a web application that tracks productivity (which can be personal or work-oriented). [Figure 39] The inputs mark accomplishments, while the outputs invoke an awareness of your social circle's productivity in order to influence your own. This system lets a user assemble an environment to monitor and reflect on her own and other's productivity. [77]

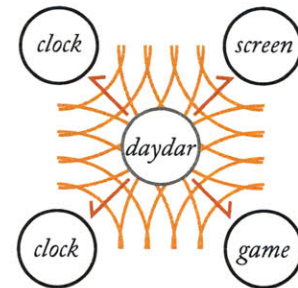


Figure 39. System diagram. Daydar follows all artifacts' activity, and sends direct messages to give users feedback through the artifacts.

3.3.4.1. DoingDoing: socially aware alarm clock

One physical artifact is DoingDoing, a socially aware alarm clock with seven bells that can be individually actuated. [Figure 38] It listens to the activity of the user's social group through the Daydar feed. When she hits snooze, a bell pings once every time someone else in her group wakes up and turns off their alarm, or checks off a to-do item. Hearing the activity nudges her out of bed, or if it's quiet, she can laze in bed knowing that she's not falling behind in the rat race. When she turns off the alarm and gets up, other DoingDoings that are in snooze mode will ping to herald it as her first achievement of the day.

DoingDoing is built using a Finger module that rings solenoid-controlled bells when it receives commands from a Hand that's connected to the Daydar application, and relays back user input (alarm turned off, the user is out of bed).

- *Input:* keywords *buzz* and *buzz <address>*. These ring a random bell, or a sequence of bells (numbered "1" through "7").
- *Output:* "snooze pressed", "alarm armed", "alarm disarmed".

Scenario

Daydar describes a future ubiquitous computing application within which various physical and digital agents work together to help the user to monitor their own and others' productivity, get motivated, and document and visualize the process of whole projects. We made a few working agents, and the rest are display mockups. The following narrative illustrates the sort of rich human-centered experience that product and interaction designers are wont to create, given the right tools.

At home



Figure 40. Toast printed with personal productivity metrics.

In Boston, Julie's alarm clock goes off at 6:30 a.m.—as usual, she was optimistic about getting an early start. She reflexively hits the snooze. But instead of shutting up, it begins to sporadically 'ping' as friends in Europe are already up and checking off to-do items. It takes several of these gentle prods before she gives in and gets out of bed. She hits her alarm clock a little harder than she needs to, hoping to make some other sleeper suffer for it.

She gets her breakfast from the toaster and glances at the information printed on it. [Figure 40] One slice shows her the types of needs she's met lately—the words 'Social' and 'Physical' are large, and she can see that Intellectual needs in particular are being neglected. The other slice shows her something off her Intellectual to-do list that she can do before she leaves for work—take a book that she's been meaning to finish, to read on the train.

At work



Figure 41. Social productivity radar on desktop.

When Julie gets to her desk, she glances at the background on her computer desktop and sees from the size of different arcs sweeping around a clock that much of her social workgroup has already started checking off items on work-related to-do lists, not just in Europe but in the US. [Figure 41] She cleans off her desk in order to quickly get “on the radar” of her social group.

Then she glances at a small whiteboard-like display on the wall that shows her the top priority right now, “Edit research paper,” and two alternate tasks. These are chosen from her to-do lists by context (she's logged into the system from a network address at work), self-weighted priority and due date entered from Remember the Milk, matching estimated time against schedule to help ensure

that she'll have enough uninterrupted time to get in the groove and finish the task. After half an hour of reading source material to possibly reference for her paper, Julie's mind starts to wander, but the whiteboard is there to remind her of her goals. She does one of the short alternate tasks as a break, then goes back to the main task and gets satisfaction from wiping it off her digital whiteboard with a gesture when she's done.

It's 45 minutes to lunch, and after getting a big task done Julie is tempted to idle for a while. But a challenge from Jay in New York pops up on her whiteboard, with a reward of a shared virtual lunch break. She sets the timer to 45 minutes and agrees by tapping on the whiteboard. Her whiteboard turns into a game clock and gives her both quick tasks and more involved tasks that count for more. Each time she finishes a task, it takes a picture of her with the webcam and sends it to Jay's whiteboard to taunt him. [Figure 42] Her strategy is to fire off as many email-related tasks as she can to intimidate Jay upfront, then see if she can fit in the sketches for a website layout for a big finish. She gets 5 emails sent, but doesn't quite finish the sketches to her satisfaction. But she got a good start on it that she'll finish after lunch, and she still beat Jay. With a feeling of accomplishment, they've earned an IM lunch chat.

After work

It's after 5:00 p.m., and Julie can see the work arc on her desktop radar tapering off as the others leave to go home or get drinks. She works a little longer since she's in a groove on her project, and cognizant that she can skip the gym and socializing for a change. Around 6:00 p.m. she logs off and heads to the train station. When she pulls out her phone to idly text and listen to music, the phone notes that she is no longer near her desk and suggests that she do some thinking about one of the items on her creativity to-do list, picking a home improvement project to tackle for the summer.

When Julie gets home, she gets on her computer to organize her thoughts about the kitchen retiling she's decided to do. She adds "Kitchen retiling" to her task manager, which then encourages her to break down what seems like a big task into manageable chunks by entering the primary subtasks involved, and then repeating the process with each of those, until she's fleshed out enough tasks with short time estimates that she can start tackling now.

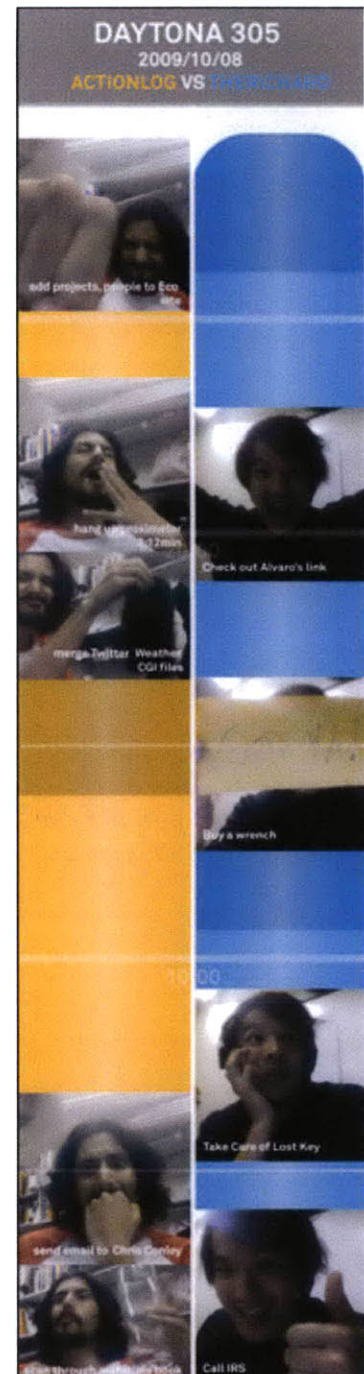


Figure 42. Motivating task completion with a social game.



Figure 43. Actuated hula girl, notification dashboard object.

3.3.5. Connected dashboard

This concept, executed with the support of IDEO, is designed to improve communication within the Zipcar car-sharing service's growing and heterogenous user community. It specifically addresses the friction that arises from a driver returning a shared car late, with carrots rather than sticks. The company's existing policies are punitive, which is at odds with its playful image.

The warmth and reach of physical and social networking channels are used to amplify the sense of community among a group of strangers, through the vehicle they share. Zipcar playfully names its cars, and this concept gives them a voice as well, through Twitter and expressive dashboard objects. It allows drivers to get, and send, the right information at the right time and place in order for the whole operation to flow smoothly—in a friendly, human manner.

The question that drives the interactions is: For this driver, and the next, what is the right information to have at any given time? I broke it down into the one discrete piece of information that's important now, and an ambient chunk of general data. The former is conveyed through the hula girl in the car, and a Twitter account away from the car. The latter uses the happiness gauge in the car, and the Twitter profile away from it.

How it works

An interaction prototype of the system is built. The hula girl is modified with a motor, cam, and a Finger to be connected to the Zipcar's onboard modem. [Figure 46] It can be commanded to wiggle in preprogrammed patterns, which were authored with Baton. It will report when its base has been tapped, and what the Hand posts on Twitter depends on context. For example, if the base is tapped right after an alert was sent to let the driver know that her time is almost up, it means she's requesting a rental extension.

The happiness gauge, prototyped in video, would be similarly built, following and reacting to the feeds of drivers and Zipcar's system. The social hub is a software agent built with Hand/Fingers that connects the other two components to the car's Twitter account.

This project called for both object-object and human-object communication, providing a feedback channel through which people not in contact with the connected objects could participate. [Figure 44] As a popular service and a flexible protocol, Twitter is an ideal medium for the application.

3.3.5.1. Social hub

Each car has a Twitter account. This acts as a messageboard of sorts that encourages responsibility in the car's driver micro-community. Sensors in the car and Zipcar's systems log events such as when the car is checked out and returned, damage is reported, the gas tank is filled, and so on. Regular drivers can follow their car's activity through Twitter, direct message it to make a reservation, and post trip journals from the car's account when they have it checked out.

- *Input:* Twitter direct messages containing keywords "waiting" and "ready?", "okay" and "yes", and "no" and "nope".
- *Output:* Twitter posts containing status updates like "I'm back home", "I'm running late", "I'm ready".

3.3.5.2. Happiness gauge

This is an ambient display in the instrument panel that aggregates the micro-community's neighborliness, treating it as a view into the car's state, like the fuel gauge or tachometer. [Figure 45] Events that push its needle toward Happy include a fill-up at the gas station, a clean car, and posting a trip journal. Sad events include a

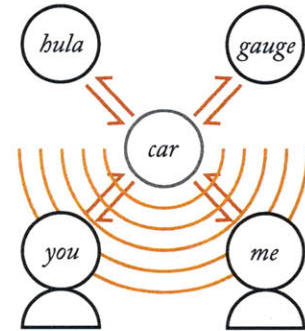


Figure 44. Social system diagram. The car posts updates on activity for followers, and its drivers can negotiate scheduling with it. The agent handling the Twitter account then interacts with the physical artifacts.



Figure 45. Happiness gauge, ambient dashboard object.

late return, complaints about being dirty, or if a driver is reported smoking in the car. These are parsed from the car's Twitter account.

3.3.5.3. Dashboard mascot

This takes the form of a hula girl or other tchotchke to provide timely alerts and reminders. [Figure 43] It addresses two friction points in the Zipcar experience with a warm interface to soften negative experiences.



Figure 46. Hula girl with motor and cam to make her shake on command.

The first is a dirty car left for the next driver. The hula girl wiggles at the beginning of a reservation to remind the new driver to check the condition of the car. If the driver is satisfied, he taps her base, and she nods a curt acknowledgement while logging the politeness of the previous driver.

The second is a late return without notice. The hula girl wiggles to remind the driver when his reservation is almost over. He can then tap the base to ask for a 30-minute extension. The hula girl will then either wiggle a curt acknowledgement or a frenzied rejection. In either case, Zipcar and other drivers following the car on Twitter will be made aware of a late return.

- *Input:* keywords *wiggleAlert*, *wiggleYes*, *wiggleNo*.
- *Output:* "I've been tapped" when the base is pressed down.

Figure 47. Tableau memory-sharing nightstand.





Figure 48. Checking the drawer for photos.

3.3.6. Tableau

Remember when we made a connection by handing someone a photo? Now we fiddle with too many cables, menus and communication channels, and the person on the other end fades away. Can we return to sharing experiences intimately while retaining the power of digital communication?

As our social circle spreads across a wider geographic area, we look for meaningful ways to share experiences. Technology has made this possible through connected objects such as computers and digital photo frames, but the interfaces for doing so are not as intimate or simple as handling physical media. My great aunt Olga loves writing letters and shuffling through photos. I, on the other hand, write emails and share photos on Flickr, and as a result, we don't communicate nearly as much as we'd like.

Tableau is an Internet-connected nightstand that stores and retrieves memories using a Twitter account. [Figure 47] It is a bridge between users of physical and digital media, taking the best parts of both.

User experience

Tableau quietly drops photos sent to it through Twitter into its drawer, for the owner to discover. [Figure 48] Images of things placed in the drawer are posted to Twitter as well. A softly glowing knob that almost imperceptibly shifts color invites interaction without demanding it. Rather than demand the owner's attention like most electronics, the knob gradually lights up green over several days to indicate that something new is in the drawer. A violet light indicates that it needs to be fed more paper.

This nightstand provides a humane digital interface, overlaying new functions onto established interactions rather than shifting computer interfaces to new contexts. The drawer becomes a simple interface to a complex computing experience, which now fits into the flow of life.

Technology

Tableau integrates a photo printer, camera and wireless Internet connection with a refinished heirloom nightstand. The drawer has been given an RGB LED knob and a sensor to detect when it's opened, controlled by an Arduino running Finger software. A Nokia Series 60 smartphone running an application written with the Hand framework routes data between the Finger, the printer, and Tableau's Twitter account.

When its Twitter feed contains a link to an image, that image is printed by the integrated photo printer and dropped into the drawer. If an object is placed in the drawer by its owner, an integrated camera takes a picture which is posted to Twitter. [Figure 52]

The integrated printer uses the Zink thermal printing process, which allows the print engine to be compact and reliable, and uses no ink. The only consumable is thermally-activated photo paper, which is dropped in a small rear drawer containing the paper feed tray. [Figure 49] This greatly reduces the complexity of maintenance.

Heirloom electronics

Tableau was preceded by an earlier design concept in illustrating the interaction of a bridge between digital and physical media. The



Figure 49. Rear drawer for paper.



Figure 50. Vivien standalone photo printer concept.



Figure 51. Original materials left exposed.

Vivien photoradio is a standalone wireless printer that receives emails with image attachments as picture postcards printed into a photo frame. [Figure 50] I used it to explore the value of using existing physical modes of interaction in digital communication.

Tableau further refines this into an anti-computer experience. Both the physical and electronic features reject obsolescence. The table is an old one, and when it was refinished, the wood of the drawer and a core sample of previous finishes were left exposed to celebrate its history. [Figure 51] A sort of patina for digital media is suggested by the green “pixel moss” motif on the tabletop. The trappings of electronics are removed except for a vestigial cable “tail,” serving as a knob for the paper drawer, that signals what lies beneath (the cable is cut because the table is wireless, of course). [Figure 49] The resulting object encourages a deeper emotional attachment that consumer electronics do not have.

Tableau’s physical characteristics are indicative of a long-lasting heirloom object. This would be misleading if it did not have digital affordances that allow it to be reprogrammed and repurposed. Thus, its basic functions are accessible through its Twitter account:

- The knob can be asked to change color.
- The drawer reports when it is opened or closed, and can be asked for the last time that happened.
- The camera can be asked to take a photo, to which a URL is posted on the nightstand’s Twitter account.
- The printer can be asked to print an image given a URL.

In this way, the interactions described are written for Tableau completely through Twitter messages. Without requiring hardware hacking (but needing some coding skills until better tools come along), a user can write an application residing on Twitter—for example, one that prints a newsworthy photo from the Reuters news feed daily. Any user can then “install” it on her Tableau by following the application as she would a person on Twitter. This wards off the obsolescence that typically befalls closed-off turnkey electronics.

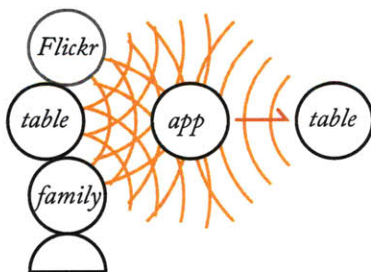


Figure 52. Social system diagram. The table can post pictures in its drawer through its app. The app also filters online album activity, and humans posting photos through their tables or manually.

4. Synthesis

Discussions around ubiquitous computing sometimes reach to the abstract. Computing is a vapor, invisible and everywhere. This is not a useful concept in understanding how to build ubiquitous computing systems, however. Computers will continue to be discrete *things* for a while yet, just smaller and more plentiful.

Where we are now is that consumer electronics are the next step down this path after personal computers. They're still shiny plastic and metal objects that we describe as "high-tech," meaning "complex." Yet most of the things in our environment are not. I'm sitting on concrete steps in front of a wood-and-glass door leading into a room with fabric-covered seats and ceramic coffee mugs sitting next to paper magazines on a wooden table. Like Weiser's eyeglasses, these are all technology that's become invisible.

How do we reconcile our current idea of computing with the objects that already are ubiquitous? The Context chapter outlined the neighboring boundaries of these two things. The Exploration chapter documents my experiments with pushing those edges closer together. Here I have collected principles learned from those experiments, divided between the networked and physical interfaces of objects. These outline my vision of designed computational objects.

4.1. Social

Like many engineers, I've built custom systems of objects to create a desired interaction. This is typically a laborious process as one has to build custom objects and the links between them to even start testing that idea. What if the inputs and outputs of objects were exposed with no particular forethought as to how they would be used? It would free a designer to create interactions more rapidly, with objects they may have not originally programmed.

Better yet, it can make more people designers. I propose using a social network as a system bus for embedded computing devices. This greatly lowers the barriers to creating and managing ubiquitous computing environments. Network topologies are a complex topic for the average person, but that same person is likely to have managed very sophisticated networks of people that they live and work with, most lately with social networking websites. It allows any Twitter client to be used as a system terminal, and may also lead to better tools built on top of a social network API, ones that are tailored to the needs of end users managing their devices.

Social object is the term I use to describe the basic unit of ubiquitous computing. In object-oriented programming, it would be called an abstract class. This class requires only that the instances of it expose their functions, and can talk to each other over a common bus. (For the experiments in this thesis, that bus is Twitter.) It's an OOP concept, with the difference that these are (mostly) physical objects. Addressing each other is not as simple as when they reside in a shared memory space, so connectivity becomes a de facto requirement.

With the Hand/Fingers toolkit, one can rapidly make previously mute devices social by adding a few lines of code. The inputs and outputs of an social object are exposed to the network by marking functions to be made available. Upon request, it will return a list of the commands it understands, and a message that contains any of those commands triggers the corresponding function.

This makes it possible to configure object networks out of devices without getting into their code—essentially, applying an object-oriented programming model to physical objects that expose an

API. In a model-view-controller pattern, [65] the physical objects are views, the user-facing component. Hand routing applications are controllers. Web services with Twitter interfaces (which could be added using Hand) serve as the model, providing data. Twitter messages are persistent and so also provide data storage.

The experiments I built with the Hand/Fingers toolkit leverage the technical work put into Twitter. Ease of configuration is the goal, and the Twitter system provides key features that enable flexible inter-object applications that would be difficult to do with other tools. I'll describe the features of an object ecosystem and social objects in this section.

4.1.1. Bus

The first step is to make devices more chatty—e.g. they should announce every affordance action and every state change. They should listen for messages to cause them to exercise their functionality. They can operate on several scales simultaneously: an action could generate a bunch of tweets that describe the effect of that action as well as a macro level tweet that indicates the action itself.

Once all of our devices have voices, we need to open the lines of communication. The core idea is to extend the system notion of a bus, where various devices can speak and listen. Inside a system, the bus can be an electrical reality, but if we want to extend this, it needs to be a virtual bus. Twitter, or at least its protocols, are well-suited as a system bus to interconnect a ubiquitous computing world.

4.1.2. Addressing & filtering

Twitter is flexible enough to handle both broadcast and point-to-point messages. But like conversations in the physical world, most Twitter communication can be overheard. With broadcast protocols, it can be easy to get flooded by irrelevant messages if you don't have a tuner. To help the receiver filter the chatter, the sender can employ a number of addressing methods, which are expressed through well-understood types of social relationships (and the Twitter constructs that enable them) [Figure 53]:

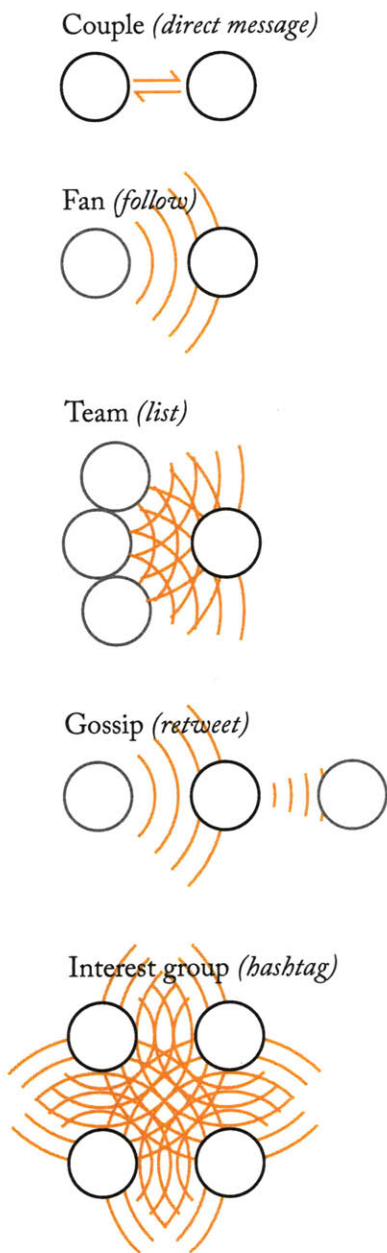


Figure 53. Relationships on Twitter mapped to social objects.

- A. *Couple (direct messaging)* – The most common relationship is a pairing between two devices. These may complement each other in function, such as Google Calendar and an alarm clock, or be identical objects in different contexts, like matching digital photoframes. This is a point-to-point conversation—no other objects can see a direct message—and so it’s very similar to the way many existing protocols work.
- B. *Fan (follow)* – A more interesting relationship is one where an object follows a number of other objects which don’t necessarily reciprocate. Examples include the wallet that follows bank accounts, or the Orb that follows Grandma’s kitchen appliances.
- C. *Team (list)* – This is a way to structure objects being followed into task-specific groups, set of hierarchal relationships between an agent and a group of objects that it observes or commands to achieve a goal. Lists are typically represented by “@username/listname”. A list is not necessarily a subset of all followed objects, and may be viewed by other objects. Objects with multiple functions, such as light bulbs, could use this to simplify management for their owner.
- D. *Gossip (retweet)* – Sometimes an object may want to spread messages it receives in order to manipulate the behavior of other objects. This may act as a filter, such as an agent that forwards only important alerts from other sources to the alarm clock while its owner is sleeping.
- E. *Interest group (hashtag)* – This is a loose grouping of objects that share an interest in a certain kind of information, decentralized into many objects. Resnick’s simulations with turtles and termites [7] suggest potential applications, one being the Watt Watchers connected light bulbs. It represents metadata for the message: #video or #photo for file type of embedded URLs, #finance or #energy for categorization.

In addition to these constructs, a search API that can match keywords in content or metadata is available. The social relationships that Twitter uses for addressing are useful to filter the noise as well.

4.1.3. Discovery

Merely listening to and processing the output of other social objects is a useful thing by itself. But to create a fully interactive object ecosystem, each object needs to be able to explicitly send input to remote objects. There is no standard API for this yet—the range of possible application domains provides a vast space to cover. The experiments covered in this thesis begin to explore the space, but it may not yet be practical to formalize a vocabulary. For the time being, API discovery is provided.

Each social object has functions which can be accessed to provide it with input or query it for output. When using the Finger code library, a developer marks the functions to be exposed. If sent a blank message or one that can't otherwise be interpreted, the object will return a list of functions it understands, much like a UNIX command line tool.

This is an apt comparison since complex functionality in UNIX-like systems is built upon many such tools. The output from one self-contained program is “piped” to another, and many of these can be connected into a longer pipe to accomplish things that the tool developers did not preconceive—in line with the idea of a social object. At least within the world of UNIX command-line users, this has proven to be a popular system even without a strictly defined set of interaction rules.

Each social object has a profile on Twitter, where information about its capabilities may be stored. This may include its API, but also its owner, location, and roles. This will be important as apps must make sense of a variety of ecosystems. An app looking for the house's alarm clock might query within the owner's objects for ones that understand “alert,” and narrow that down to ones that are listed as being in a bedroom. Or the app might look for an object that the user has tagged with the role of “alarm clock”—which may not even be a physical alarm clock, but some other object that knows how to “alert.”

4.1.4. Access Control

In “The Computer for the 21st Century,” Weiser notes privacy as a potential problem in a world where the density and connectivity of devices is high. [20] We must strike a balance between openness and simplicity. The value of the network increases with the number of possible connections, but a node must have rules to prevent unwanted connections without direct user authorization. And too many controls will cause the user to neglect security, leaving the network closed and less useful or open and insecure depending on default settings.

Twitter itself has two primary access control mechanisms that are used to protect privacy, specific to an account or to a single message. Accounts can be “protected” so that only accounts that are specifically approved can see its messages. A direct message ensures that only the addressed account can see the message. In both of these cases, a message may be relayed by the receiving account through a retweet.

This feature can be used to create software agents that selectively route messages, providing more nuanced access control. A user can choose to follow (and be followed by) an app that employs a particular security scheme. The app could route only messages from accounts certified by a trusted authority, or using the social graph, only messages from friends of friends.

4.1.5. Persistence

Unlike on many low-level transports, messages do not disappear once they are delivered. Twitter stores all messages and provides hooks to retrieve these by time, keyword or account. This allows it to be used as a form of network storage, especially with links to media on the web—useful for devices with limited resources.

4.1.6. Limitations

While I use Twitter to execute experiments and to anchor a discussion about device networks, it has some drawbacks. Chief among these are security and reliability. I have already discussed access control between devices. However, Twitter is controlled by one company, and all messages pass through its servers.

StatusNet [27] is an open-source alternative. It is compatible with the Twitter API; an existing client can be switched by pointing it to a different proxy server. With this, users can maintain control over the conversations of their devices by having their own server. StatusNet may be useful for applications where more control is needed, such as a closed household network, but it may discourage heterogenous networks of trusted and untrusted devices.

Ultimately, I believe that Twitter is a good public platform for social object development at this time. StatusNet demonstrates that API-compatible platforms that are engineered for specific purposes may be created to interoperate with or replace Twitter as the bus with minimum disruption at the client.

Figure 54. Furniture that walks is not honest. “Evil Sits Down for a Moment,” The Tick.



4.2. Super-mechanical

One important advantage of [tangible user interface] is that users receive passive haptic feedback from the physical objects as they grasp and manipulate them. Without waiting for the digital feedback (mainly visual), users can complete their input actions (e.g. moving a building model to see the interrelation of shadows).

—Hiroshi Ishii, *Tangible Bits: Beyond Pixels* [74]

In the section titled “The burden of pushing/pixels,” I discussed the frustrating step backwards we’ve taken with computers. The lack of a native physical interface causes more friction than we care to admit. But how do we realistically make physical personal computing? My answer is to make the objects in our everyday lives *super-mechanical*.¹

While computer companies are trying to shoehorn the complexity they know into new forms, I propose starting with familiar objects and mapping digital functions to their physical affordances. Computer interfaces today require translation between hand and

¹ Term courtesy of David Carr.

eye and button and screen, while a firmly tangible interface provides a tighter feedback loop through passive physicality. [21]

Passing data between physical and digital representations is only one (important) characteristic of super-mechanical objects. I envision designed objects that, through their form and interface, seek to shed the negative connotations that electronics now carry. Embedding computing into the objects already in our environment can be invisible computing as Weiser saw it: "...playful, a building of foundations, constant learning, a bit mysterious and quickly forgotten by adults."

4.2.1. Evolutionary

The more context awareness you have, the less user interface you need.

—Michael Bove [80]

We used to associate seven to ten random digits with a person, but we got that portion of our brain back with the invention of the cellphone address book. Humans demonstrate an amazing capacity for learning arbitrary interfaces, but making them more intuitive nevertheless allows us to spend more of our cognitive effort doing the task. So we've developed the desktop metaphor for computers, a visual language of push plates and handles for operating doors, and the task-based automatic transmission that abstracts away the gears.

Being specialized, familiar objects, super-mechanical devices carry a lot of context—cultural, spatial, situational—that can be used to create a strong association with the super-mechanical higher function. [54] A wallet has a closer relationship with money than does a general purpose device such as a cellphone. This allows a Proverbial Wallet to drastically reduce the interface overhead to get personal financial information. The wallet tightens as you open it, representing a tighter budget. The wallet expands in your pocket to convey your overall (financial) assets. The wallet buzzes to alert you of account activity. (This technically violates the passive interface of the wallet, but it accurately conveys the dynamism of virtual money in your bank.)

4.2.2. Honest

A danger with adding intelligence to dumb objects is that you create a usability minefield common to electronics. If the machinery between cause and effect is too mysterious, users will get anxious and lose confidence in the device's reliability. This too often doesn't have anything to do with the actual reliability, but with the expectations of the user.

Super-mechanical devices are honest objects. They retain the interactions of the original object, and any additional functionality should not disturb the essential experience. One way to go about this is to overlay the natural affordances of the object with the digital interface. In particular, the smart object should be as animated as the "dumb" version. Thus, the Proverbial Wallet has a hinge that resists the user's action passively, while the DoingDoing alarm clock actuates as a normal alarm clock does. Nervous furniture, discussed in Section 4.2.4, illustrate this exactly by subtly adjusting the amount of give that they present in response to the user's weight. And each chair responds to the user in a different way, so the right chair/information pairing must be selected. If objects stray too far from their true selves, we're back to causing anxiety in users. [Figure 54]

This also means that affordances may indicate the enhanced nature of a super-mechanical object. The Tableau project experiments with this in varying amounts, from a subtle digitally-inspired graphic treatment to an color-shifting LED drawer knob. (LEDs are high-tech, right?) Most overt is the knob for the rear paper drawer. [Figure 49] Being the end of an old printer cable, it signals that this may be similar to known computer peripherals, and wireless (true, though it's not a functioning antenna). But the upright curl also gives the object a friendly tail, which positions it far away from other printers. Several of my explorations also use digital fabrication techniques, such as the precise lasercut patterns on the Proverbial Wallets or the computer-generated graphics etched into the proximeter, to root these objects in the Information Age.

4.2.3. Post-electronic

Electrical power is inevitable in a networked object, but we can work to minimize its negative characteristics. This is an emotional challenge as much as a function one. Tangible interfaces trade on the expectation that physical objects will preserve their state or change with generally understood constraints. [74] But electronic elements such as screens and LEDs convey impermanence, creating possible anxiety. (Have your parents ever expressed frustration that an on-screen window “disappeared”?) These are of course useful for many things, and will not fade away any time soon. But in my explorations, I have taken up the challenge to rely on alternative feedback mechanisms, as discussed in “The burden of pushing, pixels: rich interaction design.” When an electronic display is necessary, it can be softened. The Revo Heritage radio’s screen disappears when not in use. [Figure 7] The proximeter’s pixels are distorted by layers of glass and vellum to give them depth and soft edges. [Figure 35]

As part of the move away from overt electronics, super-mechanical devices use as little power as possible. Nothing produces frustration like a dead battery or a misplaced power adapter. There are thresholds at which the relationship changes. Every month, charge your Kindle. Every year, clean the heat exchange coils on your refrigerator. Every two years, replace the battery in your wristwatch. Even better is a watch with a wind-up mechanism. The ideal super-mechanical device is battery-free even if it is electronic. The wireless self-powered switch [71] harvests enough electricity from the use of a button to operate. Likewise, the wallet could generate energy each time its hinge is opened.

Do any of these characteristics sound familiar? Super-mechanical objects populate the world of Harry Potter—the media player that feels like a newspaper, the videochat service that looks like a pair of mirrors, the annotated Tivo in the form of binoculars. They’re all invisible computing, wrapped in a playful presentation that salutes the user and the machine. Call it practical magic, sufficiently advanced technology, or super-mechanical—I believe that this makes good tools and good computers.



Figure 55. Implicit communication through touch.

4.2.4. Nervous furniture

“That couch is a jerk.” Why would the designer make it a jerk? (Because he's a jerk.) Who would buy it? (People who would date him.) Is the couch trainable, like a dog? If you spoil it at first, how will it act later? Will it walk all over you? Will it get on the dog when you're not looking?

If objects you use every day become connected, there's a question of their behavior. Should they be omniscient or subservient? If they are too forward, they run the risk of being obnoxious. What follows is an exploration of how super-mechanicality might be applied to connected objects.

Our physical presences speak to each other without active communication. [Figure 55] How can I be present to my partner in the same way when we're not physically together? Transmitting presence should be subtle—it's enough to know that you're sharing some environment, even if it's sitting at the computer. It can be as low-bandwidth as being online and available for instant messaging.

Next to clothes, the most intimate objects throughout our waking hours are seats. They seem suitable to transmitting presence and more subtle physical cues. A Nervous Chair would read its owner's stress level through posture, fidgeting and galvanic skin response. When paired with another, the chair communicates the owner's

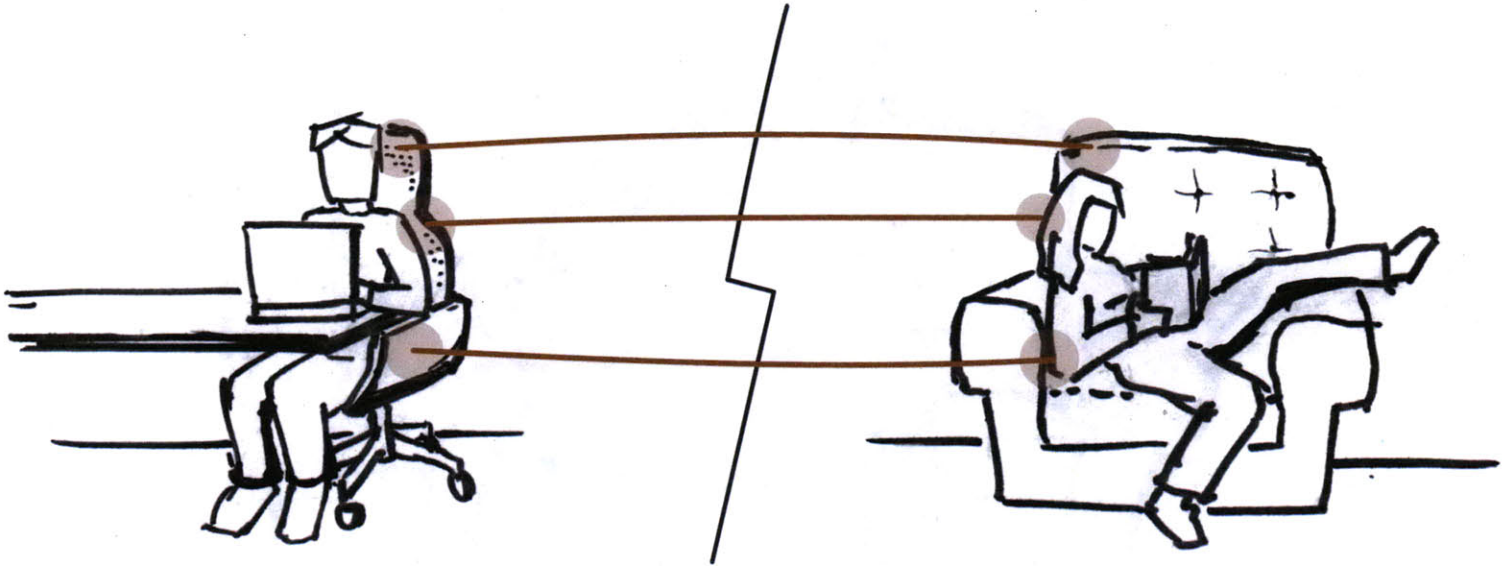


Figure 56. Remotely connected furniture.

stress level to its counterpart. [Figure 56] Sympathetic furniture could convey presence through the pressure of another body pushing back against you; transmit the ambient rumble of an office when you're working at home; or become stiff or relaxed to reflect order or disorder in your shared schedules.

It's important to communicate these qualities while remaining true to the object. Ken Perlin demonstrated how a little pseudo-random movement can turn an image into a presence. [40] I don't believe that we should literally animate objects, Beauty and the Beast-style—things behaving in alien ways add their own kind of anxiety. Instead, there are passive characteristics that can be varied in their response to the human interaction. Let each chair speak to you in its own dialect of the language of chairs that you understand. [Figure 57]

An office chair communicating relaxation will have some play in adjustments. Stress makes it tighten.

A comfy armchair responds to calm by being squishy and surrounding you. Stress leaves it deflated and uncomfortable.

A wooden shaker chair will feel solid and reassuring when calm. Stress loosens the fasteners, making it wobbly and unsure. The screws fall out. Like a rider on a skittish horse, you must be steady to control the chair's movement and tighten it, and transmit that assurance back to your partner's chair.

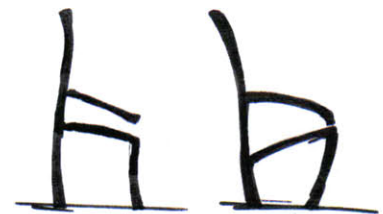


Figure 57. A chair's posture can convey stress or consolation.

4.3. Social object apps

Many consumer electronics devices could be social objects. But so can all the tangible things I describe in my house. With connectivity, the sofa knows when it's being sat upon, the door knows when it's opened, the mug knows when it's empty or being held. Just like software objects, these things have bits of info that are not useful unless they're communicating within a larger system, with controllers that know what to do with that information.

And just like a typical software-only MVC application, the designer creates social object applications by adding logic to the controllers to call functions on physical and digital objects based on output from other objects. These controllers, software agents interacting with devices through a Twitter account, are a class of social object whose role is to route the messages of other objects. Borrowing a term that smartphone platforms have made popular, a social object app is a single-task software agent that a user or device can follow in order to gain its service. The app will then follow back, in order to listen to the device's messages and to address it directly and privately. Though the app resides on the network, this process approximates installation of traditional computer applications.

The old consumer electronics model has seen multiple services converge into one object. I believe that the pendulum should now swing in the other direction, where multiple objects converge into one service. This is task-centric computing, in which the interface is distributed across connected data sources and physical objects. Apps that run in the cloud can weave available objects into environmental I/O, giving users computing experiences that fit into the flow of life. It's worth keeping in mind that each object does not not exclusively belong to that service, but is called into action by whatever associated service needs it.

Our device interactions are now tightly defined by manufacturers. social objects will make defining our own interactions easy, and defining new sorts of rich interactions possible. We may each possess an object ecosystem, a collection of social objects accessible to each other on a network, and able to form new connections. If

social apps are the DNA of multi-object organisms, an object ecosystem is the primordial soup from which it draws.

The following scenarios illustrate apps constructed with object network topologies described in the “Addressing & filtering” section. They are largely composed of social objects that I have developed, discussed in the Exploration section.

4.3.1. Independent living

My elderly grandmother is fiercely independent, and she lived alone for a long time. Of course, her family worries about her—whether she’s eating well, in good spirits, taking her medication—and eventually she had to move in with her daughter for peace of mind. What if her activities could be monitored remotely while affording her privacy, through her objects?

The Watt Watchers project used connected light fixtures to dim the lights in a home when enough of them were turned on, as an ambient energy consumption reminder. If each light had a Twitter agent, this could be done by having them tweet “I am on #home-lights” or “I am off #home-lights” whenever an occupant flipped a light switch. Each would also be watching for tweets with the #home-lights tag and keep a count of how many others were on at the moment, and at a certain threshold, dim to notify the occupants that several lights were on in the house. [Figure 58]

This would be more centralized than the original Watt Watchers—the state of each bulb is shifted to cloud memory—but more flexible and manageable as a result. But now, having their activity logs accessible on the network could be used for other applications as well. My grandmother’s bedroom light could be set to gradually come on seven hours after all the lights have been turned off, or mimic summer daylight hours during the winter. Or her children could see the usage of lighting in her house to abstractly monitor her activity. The lighting could represent sunlight to a virtual plant on their photoframe which is following the @grandma/lights list. [Figure 59]

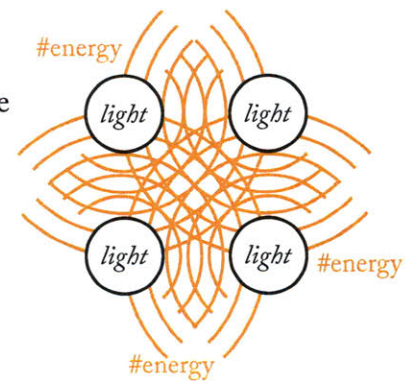


Figure 58. Watt Watchers implemented in social objects.

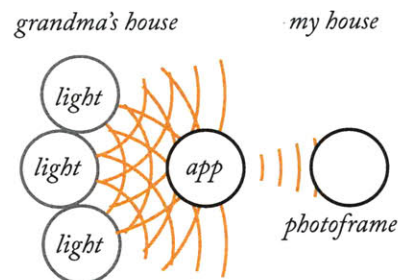


Figure 59. Abstract representation of remote activity.

4.3.2. Finance

Many people have difficulty in grasping their financial situation as it has become disassociated from physical money. Daniel is one of them. His connected wallet is the interface through which he can better control his assets. It follows the private tweets of Daniel's bank accounts. When an account tweets activity, the wallet may reflect transactions by vibrating, or reflecting overall balance in the resistance of the hinge when opened (and tweet back out the fact that it was opened). At home, Daniel places the wallet on the desk in the home office with an integrated display, which pulls up all the tweets the wallet is generating and following, and visualizes them by category to provide insight into his spending habits. [Figure 60]

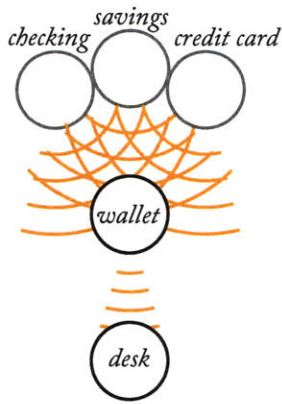


Figure 60. Proverbial Wallet implemented in social objects.

When Daniel covers dinner for his friends, he taps his wallet to theirs. They all have an app that tracks who owes whom. His app learns from his wallet that he paid this time, and it sends a tweet to the friends' corresponding apps, tagged with #paidfor for future reference. [Figure 61]

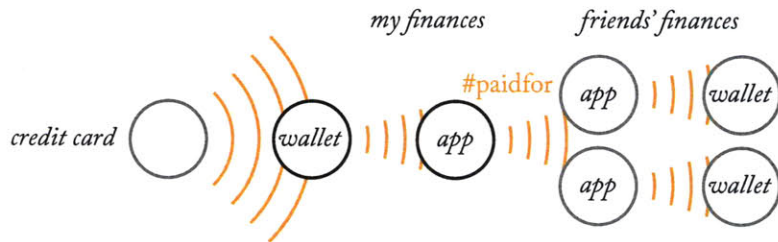


Figure 61. Social app for Proverbial Wallets.

4.3.3. Sharing memories

Natasha empties her pockets and purse into a drawer of her connected Tableau nightstand. It can digest the receipts, notes and cellphone snapshots of the day into a digital format. One app that she follows post mementos to her online scrapbook, which can be shared with friends and family. Her mother keep volumes of photo albums, and follows this app to print her children's most liked photos in a drawer of her own nightstand. [Figure 62]

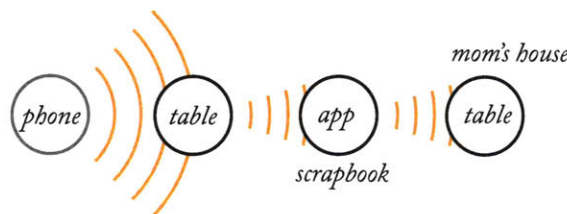


Figure 62. Sharing app for Tableau with social objects.

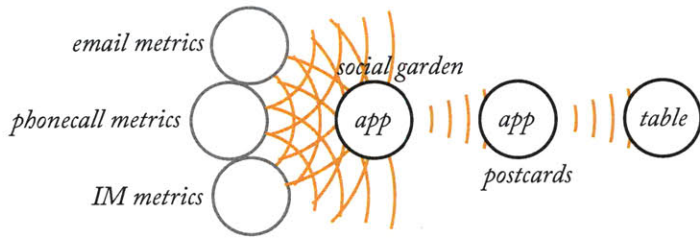


Figure 63. Social Garden reminder app for Tableau.

Another app may produce picture postcards with photos that Natasha’s taken, addressed to friends she hasn’t talked to in a while, which it figures out by asking her Social Garden app that’s tracking correspondence metrics. [Figure 63]

4.3.4. Waking up

Therese arms her connected alarm clock as she turns in for the night. The clock tweets this action. An app that is following the clock registers that, and asks her online calendar if she has any morning meetings. If she does, it will send a message to the clock to ring two hours prior. [Figure 64]

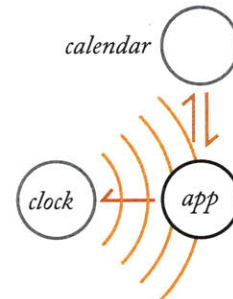


Figure 64. Schedule-aware alarm clock.

If she doesn’t have any meetings in the morning and her lights register being turned off after midnight the last few nights, the alarm takes on a social quality. In this mode, the alarm clock asks the Daydar service to report the level of productivity in Therese’s peer group, ringing a bell just once each time a friend checks off a to-do item after she hits the snooze button. In this way, she can catch up on her sleep without worrying about keeping up with the Joneses. [Figure 65]

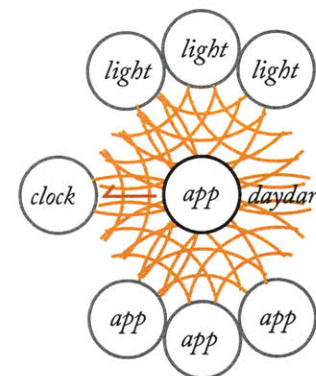


Figure 65. Context-aware Daydar app implemented in social objects.

When Therese finds a new app that lets her experience the perfect sunrise, she follows it on Twitter to give it access to her object ecosystem. Early every morning it checks with other services on Twitter for the local weather and the time of sunrise. If conditions are right and her object ecosystem indicates that she went to bed early enough, the app will ask her alarm clock to wake her up in time to experience a great sunrise. [Figure 66]

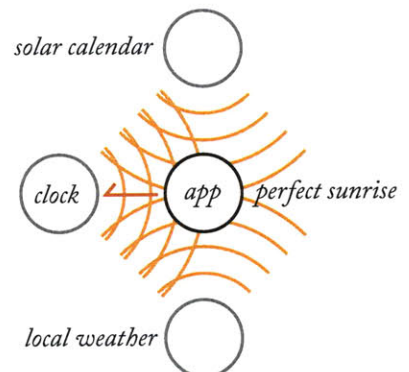


Figure 66. Perfect sunrise app.

4.4. Future work

With this thesis, I hope that I have opened a door for the field of product design to begin thinking about what future products will look like when fully network-enabled.

Twitter has been a useful tool to prototype network interactions, and will remain so for a while. But beyond that, it is a powerful metaphor for understanding the complexity of networks of objects. Twitter's success is in allowing humans to self-organize through messaging addressing methods that have existed since the beginning of civilization. I'm proposing that giving similar abilities to objects can benefit the designers of object systems.

Extending this metaphor further into realization suggests that networks of objects will start experiencing problems familiar to human societies. Will this model scale with volume? I believe there will be some degradation but that the benefits will still far outweigh the costs, just as in human society.

Twitter provides a simple centralized solution for addressing now, but for it to scale to billions or trillions of devices, a tiered addressing system will become necessary. This is already solved by the Internet's domain name system, and the separation of wide-area and local-area networks. People may have messages between their own social objects routed by a local server, powered by an open-source Twitter alternative such as StatusNet. [27] This server would also bridge the local object ecosystem and the larger "wild" ecosystem, managing permissions and addressing.

This thesis has not attempted to define a standard vocabulary for social objects. My contribution has been to open the definition of a physical product to include services distributed across a network, and to describe how this must change the interface of products. The technical work of vocabularies will start with a general specification, equivalent to XML, on which future domain-specific languages may be built.

The evolution of the Twitter service itself suggests that the proposals in this thesis will most likely take root through the participation of the network, not through specifications in an

academic paper. It began as a relatively blank slate, which drove users to create the syntax that they needed to organize, and the clients to understand it. Twitter, in turn, codified the successful syntax into the official API, which has magnified users' ability to derive utility from it with more advanced services. I believe the challenges posed here will also be best answered through an iterative loop between the toolmakers and the users. The tools I have made will be released shortly; it will then be the users' turn.

5. Conclusion

Emergent properties are new attributes of a whole that arise from the interaction and interconnection of the parts. The idea of emergence can be understood with an analogy: A cake has a taste not found in any one of its ingredients. Nor is its taste simply the average of the ingredients' flavors—something, say, halfway between flour and eggs. It is much more than that. The taste of a cake transcends the simple sum of its ingredients.

—Nicholas A. Christakis and James H. Fowler, *Connected: The Surprising Power of Our Social Networks and How They Shape Our Lives*

Regardless of how complex they are, software interfaces are additive. Starting with nothing, developers have deliberately chosen each element. The nuances of physical objects comes from a subtractive process. A block of wood comes with many characteristics, and by shaping it, a designer is only changing a few of them.

Similarly, a highly connected network of objects is raw material out of which rich software interfaces can be carved. Object-oriented programming removed a barrier to creating and maintaining large software systems, increasing programmer productivity by leaps and bounds. Electronic channels leading up to social networking services such as Facebook and Twitter have done the same for human communications.

These provide a productive model for how objects may communicate with each other, and humans. Twitter reduces hidden complexity through a human-scale model. By thinking of objects as social beings, we can more easily conceive how they might communicate. We can free ourselves to develop complex multi-cellular systems, managed through simple apps that route messages between cells. These object ecosystems are not just composed of objects connected in a network, but interdependent objects that draw on the functions of others to create nuanced interactions with a variety of simpler objects. And with an API discovery mechanism, apps will provide the rules to seed even more flexible,

self-organizing systems. Resnick points out that decentralized thinking is hard for humans, [7] but this is what is needed to make the most of social objects.

I have made an argument for manufacturers and designers of consumer electronics to adopt the features of social objects. They're closer than they know. Noted in "Machine use of Twitter" are consumer electronics that already have Twitter accounts, ostensibly so that users can let others know what show or e-book they're consuming. This information might be more interesting to social objects than to people. If the e-book reader was given its *own* Twitter account, it could leave a trail of exhaust data that is only interesting to software agents, which could use it to provide context to the user's whole environment.

The relatively small effort of giving products a Twitter account on which they post each bit of activity, and expose their essential functions, will be repaid in new markets spurred by users' creativity. Open systems allow expert users to tinker and come up with new and better applications, which we can use to ward off the idea of obsolescence, and which manufacturers can incorporate into polished products for the rest of us.

A cake may be sustenance, but someone took the care to combine the ingredients in such a way that an enjoyable snack was produced. I hope that this work provides a recipe for a future computing environment and its components, in a way that puts a humane experience at the center of it. I trust designers to create something more than the sum of these blocks. There will be snacks.

6. References

1. Polaine, A. "Talk to the Hand: Dan Saffer and gestural interfaces," Core77, 2 Feb 2009. http://www.core77.com/blog/featured_items/talk_to_the_hand_dan_saffer_and_gestural_interfaces_by_andy_polaine_12522.asp
2. Sterling, Bruce. *Shaping Things*. Cambridge, Massachusetts: The MIT Press, 2005.
3. Dunne, Anthony. *Hertzian Tales*. Cambridge, Massachusetts: The MIT Press, 2008.
4. Dobson, Kelly. *Toward Machine Therapy: Parapraxis of Machine Design and Use*. Online Communities and Social Computing, Springer Berlin / Heidelberg, 2007.
5. von Hippel, E. and Katz, R., Shifting Innovation to Users via Toolkits. *Manage. Sci.* 48, 7 (Jul. 2002), 821-833.
6. Hartmann, B., Abdulla, L., Mittal, M., Klemmer, S. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition, Proceedings of the SIGCHI conference on Human factors in computing systems, April 28-May 03, 2007, San Jose, California, USA.
7. Resnick, Mitchell. *Turtles, Termites and Traffic Jams*. Cambridge, Massachusetts: The MIT Press, 1997.
8. Hartmann, B. et al. Reflective Physical Prototyping through Integrated Design, Test, and Analysis. Proceedings of UIST'06, October 15-18, 2006, Montreux, Switzerland.
9. Weiser, M. The world is not a desktop, *interactions*, v.1 n.1, p.7-8, Jan. 1994.
10. Chang, Jade. "The Chair: Optimized lifetime." *Metropolis*, October 2007.
11. Buechley, L. LilyPad Arduino: How an Open Source Hardware Kit is Sparking new Engineering and Design Communities. CHI 2010.
12. Holtzman, H. and Bove, M. Consumer Electronics 2.0 white paper. Technical report, MIT Media Laboratory, Cambridge, Massachusetts, October 2006. <http://wiki.media.mit.edu/pub/CE20/WebHome/CE20-whitepaper-v2.pdf>
13. Pierce, J. Material Awareness: Promoting Enduring Product Relationships through Reflection. CHI 2009.
14. Arikan, Burak. Collective systems for creative expression. Thesis (S.M.) - Massachusetts Institute of Technology, School of Architecture and Planning, Program in Media Arts and Sciences, 2006.
15. Stumpf, Bill. *The Ice Palace That Melted Away*. Pantheon, 1998.
16. Arduino physical computing platform, <http://www.arduino.cc/>

17. Norman, Donald. *The Design of Everyday Things*. New York: Basic Books (Perseus), 2002.
18. Norman, Donald. *The Invisible Computer*. MIT Press, 1998.
19. Buxton, W. *Less is more (more is less), The invisible future: the seamless integration of technology into everyday life*, McGraw-Hill, Inc., New York, NY, 2001.
20. Weiser, M. "The Computer for the 21st Century", *Scientific American* 265 (3): pp. 66-75, 1991.
21. Ishii, H., and Ullmer, B. *Tangible Bits: Towards Seamless Interfaces between People, Bits, and Atoms*. In *Proceedings of CHI '97*, pp. 234-241.
22. Ishii, H., Wisneski, C., Brave, S., Dahley, A., Gorbet, *Integrating Ambient Media with Architectural Space*(video), *Conference Summary CHI 1998*, ACM Press. M., Ullmer, B., and Yarin, P. *ambientROOM:(video)*, *Conference Summary CHI 1998*, ACM Press.
23. Djajadiningrat, T., et al. *Tangible products: redressing the balance between appearance and action*. *Personal and Ubiquitous Computing*, v.8 n.5, p. 294-309, September 2004.
24. Fry, Ken. "The Experience Imperative: A Manifesto for Industrial Designers," *Core77*, 1 Dec 2009. http://www.core77.com/blog/featured_items/the_experience_imperative_a_manifesto_for_industrial_designers_by_ken_fry_15322.asp
25. Moggridge, W. *Designing Interactions*. Cambridge, Massachusetts: The MIT Press, 2007.
26. Fukusawa, N. Naoto Fukusawa. London: Phaidon, 2007.
27. StatusNet social networking platform. <http://status.net/>
28. Orban, David. *How Machines Are Going to Make Us Human, Again*. Talk at Mobile Monday Amsterdam #15. <http://www.mobilemonday.nl/talks/david-orban-how-machines-are-going-to-make-us-human-again/>
29. *LISTSERV* email list management software. <http://en.wikipedia.org/wiki/LISTSERV>
30. Norman, Donald. *The next UI breakthrough: command lines*. *Interactions*, v.14 n.3, May + June 2007.
31. "Apple //t" Twitter display, <http://www.atomsandelectrons.com/blog/post/Apple-t.aspx>
32. Milian, Mark. "Washing machine Twitters when clothes are done," *Los Angeles Times*, January 12, 2009.
33. *Reware*. <http://dev.eyebeam.org/projects/reware/wiki/Reware>
34. Louis C.K., *Late Night with Conan O'Brien*, October 1, 2008.

35. Fraser, John. Keynote, Proceedings of the 4rd International Conference on Tangible, Embedded, and Embodied Interaction, January 25-27, 2010, Cambridge, MA, USA.
36. Djajadiningrat, T., et al. Easy doesn't do it: skill and expression in tangible aesthetics. *Personal and Ubiquitous Computing*, v.11 n.8, p.657-676, December 2007.
37. Standage, Tom. *The Victorian Internet*. Berkley Trade, 1999.
38. Levy, Stephen. "Q&A: Jobs on iPod's Cultural Impact," *Newsweek*, Sunday, October 15, 2006.
39. Whipple Museum of the History of Science, Cambridge, United Kingdom.
40. Perlin, Ken. Lecture, Spring 2010, MIT Media Lab.
41. Malik, Om. "Make Your Own Personal Cloud With Pogoplug," *GigaOm*, April 15, 2009. <http://gigaom.com/2009/04/15/make-your-own-personal-cloud-with-pogoplug/>
42. McDonough, W. and Braungart, M. *Cradle to Cradle: Remaking the Way We Make Things*. New York: North Point Press, 2002.
43. Kahney, Leander. "The New Pet Craze: Robovacs," *Wired.com*, June 16, 2003. <http://www.wired.com/science/discoveries/news/2003/06/59249>
44. TrackThis package tracking service, <http://www.usetrackthis.com>
45. Quartz Composer visual programming tool, Apple Inc. <http://developer.apple.com/graphicsimaging/quartz/quartzcomposer.html>
46. Fogg, B.J. *Persuasive Technology: Using Computers to Change What We Think and Do*, San Francisco, Morgan Kaufmann, 2002, 41.
47. Knowledge Navigator concept, Apple Inc., 1987. <http://video.google.com/videoplay?docid=-5144094928842683632>
48. Woebken, C. A new relationship to digital money, 2008. http://www.woebken.net/future_of_money.html
49. Dunne, A. and Raby F. The Placebo project, Proceedings of the conference on Designing interactive systems: processes, practices, methods, and techniques, p.9-12, June 25-28, 2002, London, England.
50. Raghubir, P. and Srivastava, J. Monopoly Money: The Effect of Payment Coupling and Form on Spending Behavior. *Journal of Experimental Psychology: Applied*, September 2008, Vol. 14 No. 3, 213-2
51. Greenberg, Saul and Chester Fitchett. Phidgets: Easy development of physical interfaces through physical widgets. Proceedings of the ACM UIST 2001 Symposium on User Interface Software and Technology, November 11-14, Orlando, Florida.
52. Tufte, Edward. *Beautiful Evidence*. Graphics Press, 2006.
53. Thompson, Clive. "How Twitter Creates a Social Sixth Sense." *Wired*, July 1997, Vol. 15 No. 7.

54. Patten, James. Mechanical Constraints as Common Ground between People and Computers, PhD Thesis, MIT Media Lab, 2006.
55. Rose, David. Tangible Interfaces class lecture, MIT Media Lab, Fall 2008.
56. Kestner, John. Designers Prototyping Connected Objects, 2010. http://web.media.mit.edu/~jkestner/papers/designers_prototyping_connected_objects.pdf
57. LittleBits electronic prototyping components. <http://littlebits.cc>
58. Vitsœ. "Dieter Rams: ten principles for good design." <http://www.vitsoe.com/en/gb/about/dieterams/gooddesign>
59. Fischer, Jordan. Private communication, 2007.
60. Darlin, Damon. "Seizing the Moment." New York Times, May 10, 2007.
61. Eye-Fi camera memory card with integrated WiFi radio. <http://www.eye.fi>
62. Shelley, James. "Love Your Stuff," Inventing a Planet, June 2, 2010. <http://www.inventingaplanet.com/love-your-stuff/>
63. Bonanni, L., et al. Small business applications of Sourcemap: a web tool for sustainable design and supply chain transparency. Proceedings of the 28th international conference on Human factors in computing systems, CHI '10.
64. Foursquare social location-based web service. <http://foursquare.com>
65. Burbeck, S. Applications Programming in Smalltalk-80(TM): How to use Model-View-Controller (MVC), 1987.
66. Shneiderman, B. Tree Visualization with Tree-maps: A 2-D space-filling approach. ACM Transactions on Graphics 11, 1 (Jan. 1992), pp. 92-99.
67. Bradley, M.M., & Lang, P.J. (1999). Affective norms for English words (ANEW). Gainesville, FL. The NIMH Center for the Study of Emotion and Attention, University of Florida.
68. Hustad, Megan. "The Office Phone Call Was Music to the Ears," March 9, 2008, New York Times. <http://www.nytimes.com/2008/03/09/business/09pre.html>
69. Saffer, Dan. "Tabs, Pads, and Boards (and Dots)," November 19, 2008. <http://www.kickerstudio.com/blog/2008/11/tabs-pads-and-boards-and-dots/>
70. Steiner, Hans-Christoph. Firmata: Towards making microcontrollers act like extensions of the computer. In Proceedings of NIME 2009 New Interfaces for Musical Expression, Pittsburgh, PA, USA.
71. Paradiso, Joseph A. and Mark Feldmeier. "A Compact, Wireless, Self-Powered Pushbutton Controller," in Abowd, G.D., Brumitt, B., and Shafer, S., eds, "UbiComp 2001: Ubiquitous Computing," ACM UBICOMP Conference Proceedings, Atlanta GA, Sept. 2001, Springer-Verlag Berlin Heidelberg, 2001, pp. 299-304.

72. Owen, David. Object Lesson, "Tools," The New Yorker, March 8, 2010, p. 21.
73. Cottam, Matt. Wooden logic: In search of heirloom electronics. Thesis (M.A.) - Umeå University, Umeå Institute of Design, Interaction Design Program, 2009.
74. Ishii, Hiroshi. Tangible bits: beyond pixels. Proceedings of the 2nd international conference on Tangible and embedded interaction, February 18-20, 2008, Bonn, Germany.
75. Bove, Jr., V. M. Private communication, July 2010.
76. Polaine, A. "Interaction + Product Design: A Peek Inside the Revo Heritage Radio," Core77, 1 Apr 2010. http://www.core77.com/blog/featured_items/interaction_product_design_a_peek_inside_the_revo_heritage_radio__16303.asp
77. Kestner, J., The, R. and Holtzman, H. Daydar: framework for socially motivated productivity. <http://web.media.mit.edu/~jkestner/papers/Daydar.pdf>
78. Diener, A. "Afterlife: An Essential Guide To Design For Disassembly," Core77, 1 Feb 2010. http://www.core77.com/blog/featured_items/afterlife_an_essential_guide_to_design_for_disassembly_by_alex_diener__15799.asp
79. Ikea Hacker. <http://ikeahacker.blogspot.com/>
80. Elliott, Greg. Salvage e-waste kiosk. <http://web.media.mit.edu/~greg/salvage/documentation>
81. Instructables. <http://www.instructables.com>
82. Carr, David. Mantis CNC milling machine. <http://makeyourbot.org/>
83. Follmer, S., et al. CopyCAD: Remixing Physical Objects With Copy and Paste From the Real World. Submitted to UIST '10.
84. Kestner, J. And Holtzman, H. Hands and Fingers: A mobile platform for a person-centric network of computational objects. Proceedings of the 4rd International Conference on Tangible, Embedded, and Embodied Interaction, January 25-27, 2010, Cambridge, MA, USA.
85. Interview with Steve Jobs. D8 Conference, June 1, 2010. <http://d8.allthingsd.com/speakers/steve-jobs/>