SURFNET /

Designing Digital Surface Applications

Chapter Extract:
Greenberg, S. and Marquardt, N. (2016) Using Social Science Theory to Inspire Surface Design: A Case Study of Proxemic Interactions. In Surfnet: Designing Digital Surface Applications. (Maurer, F., Ed.). University of Calgary, Calgary, Canada, SurfNet, pages 26-38, March. ISBN 978-0-88953-388-2-(PDF).

Frank Maurer, (Ed.)

ISBN 978-0-88953-387-5 (Softcover)
ISBN 978-0-88953-388-2 (PDF)

Published by:

SurfNet

University of Calgary
2500 University Drive, NW
Calgary, AB Canada
T2N 1N4

This book has been published using funds from the Natural Sciences and Engineering Research Council of Canada (NSERC), Strategic Network Grants program.

NSERC
CRSNG

Using Social Science Theory to Inspire Surface Design: A Case Study of Proxemic Interactions

Saul Greenberg and Nicolai Marquardt

Introduction

Designers of novel surface interaction techniques and applications are influenced by many factors. Some designers follow a mostly iterative approach to system refinement, where they seek to improve existing methods by exposing and solving inefficiencies. Some try to better understand user needs such as through observational studies and by using software engineering techniques to craft requirements analysis. Some base their work around the affordances of technical innovations, where these new technologies expose a plethora of design opportunities that were not previously possible. Some incorporate advances made in other interaction fields to surface design, where methods developed elsewhere are adapted to the surface medium. Some rely on intuitions and personal experiences, where they generate ideas, sift through them, and apply, test and refine what they consider to be the best candidate designs.

Our own approach takes a somewhat different direction: we use social science theory to both guide and inspire our research on surface designs. Our basic premise is that our understanding of human-human interaction can be applied – albeit with some caveats – to human-computer interaction (HCI).

Our design process generally follows five stages. These stages are not purely sequential. All influence one another: they often overlap and may be done in parallel. Earlier stages may be revisited based upon insights garnered in later stages.

Stage 1. Identify candidate social science theories potentially relevant to surface interaction. This is by no means straight-forward. There are a plethora of social science theories, and most are of little value to aid design thinking. As well, because these theories explain human-human interaction rather than human-technical interaction, they must be read with a creative eye. This can only work if it is done actively. For every theory considered, for example, it is useful to ask "what could we do if one or more of the actors in this theory was technology (such as a large display) rather than a person?". From that question, one can then brainstorm scenarios where the designer could try to apply that theory to a design situation. Of course, this also begs the question of where in social sciences to look. Our own experiences suggests that helpful theories can be found by reading social science texts and primers introducing theories, as these are often written at a level accessible by designers and software technologists. As well, others in the HCI field may have already suggested a link between social theory and technological design.

Stage 2. Translate that social science theory into a form applicable to technological design. Social science theories are cast in their own language, with their own jargon, emphasis and interpretation. They target people rather than technology. They are rarely usable by designers 'out of the box', simply because they do not address technological innovation or design. They often include detail that cannot be applied to design situations. Consequently, it is important to recast the theory into a form that a designer can use. This could be done, for example, by simplifying the theory into its core concepts, and recasting select portions and details of that theory into a form applicable to the technological setting.

Stage 3. Quick and dirty prototyping. It is one thing to know theory, but quite another to understand its ramifications to design. Our approach advocates getting our hands dirty as quick as possible, as we believe this to be the best way to reveal design opportunities afforded by that theory. This means brainstorming ideas (e.g., through sketching), and actually building a variety of simple proof of concept prototypes that can be tried out. By doing so, the designer gains immediate feedback on the utility of the theory. If the prototypes are uninspiring, or are unnatural during use, or do not seem to resonate, then it is likely that the theory is not as applicable to design as predicted. Conversely, if the prototypes generate excitement, feel natural during use, are easily explained to others, and suggest even more prototypes, then it is likely that the theory has considerable potential to design. At the same time, the designer is exposed to the technical challenges of the domain (i.e., software and hardware development), which gives insight into tool development as done in the next step.

Stage 4. Retrenchment: Building a toolkit for rapid development. It may be (and often is) the case that applying that theory to actual systems development may require hardware that is not readily available or suitable, and/or that software development is tricky. While it is likely possible that a few prototypes can be built by brute force (stage 3), varying those prototypes can be excessively time-consuming, thus hindering the iterative process. At this point, we advocate retrenchment, where – based on implementation experiences so far – the design team turns to developing a toolkit that will dramatically simplify the programming effort of these systems. This means that concepts that are core to the application of the theory should be embedded into the system, where a programmer can invoke its features through a few lines in an application programmer's interface (the API). The primary motivation of toolkit development is to allow the designer and programmer to concentrate on the design and iteration of the system rather than its underlying plumbing.

Stage 5. Robust prototype development and full research applications. At this point, the designer should have a good understanding of the theory, along with experiences applying it to particular situations. The designer will also have a good toolkit for developing applications within the genre. This is now the time for the designer to pursue developing robust prototypes and applications, including exploring the nuances of interaction techniques. In this final stage, the designer can focus on particular problem areas and nuances within the usual human-computer interaction test/iterate cycle.

In the remainder of this paper, we will use the above stages to introduce our Surfnet project on proxemics interactions, which was built upon the social science theory of proxemics.

Stage 1. The Social Science Theory of Proxemic Interactions

In 1966, anthropologist Edward Hall coined the word 'proxemics', an area of study that identifies the culturally-dependent ways that people use inter-personal distance to understand and mediate their interactions with other people (Hall, 1966). While his theory of proxemics has many aspects to it, its most basic forms define four proxemic *zones* that characterize how people interpret inter-personal distance. While aspects of these zones are culturally dependent, western culture typically defines distances within these zones as: intimate (~0–1.5′), personal (1.5–4′), social (4′– 12′) and public (12′–25′). As these names imply, closer distances lead to increasing expectations of interpersonal engagement and intimacy. In practice, people adjust these distances not only to match their social activities, but to raise defense mechanisms when others intrude into these zones. This is something we understand intuitively, where people often adjust their positions to best fit the dynamics of their interpersonal interactions.

Hall also described how features within the space affect people's interactions. Fixed features include those that mark boundaries (e.g., entrances to a particular type of room), where people tend to organize certain kinds of social activities within these boundaries. Semi-fixed features are entities whose position can affect whether the space tends to bring people together, or move them apart (for example, the arrangement of chairs).

To understand why this theory is relevant, we need to revisit the Ubicomp vision. In 1991, Mark Weiser – recognized as the founder of Ubicomp – saw Ubicomp as technologies that disappear, where they 'weave themselves into the fabric of everyday life until they are indistinguishable from it', where computers are integrated 'seamlessly into the world' (Weiser, 1991). He envisioned many computers per person, all inter-connected, and all with varying form factors. Significantly, Weiser envisioned the day when devices would know about their location and surroundings, where their behavior and function would depend to some extent on their environmental context (we now call this context-aware computing). As time passed, modern technology is now realizing parts of Weiser's vision, what with the common use of smart phones, tablets, laptops, large digital touch surfaces, and other information appliances. Many devices also exploit location-awareness, where the combination of global positioning systems (GPS) and compass information (location) is used in tandem with knowledge about the physical environment (e.g., nearby businesses and services).

Yet Weiser's vision of seamlessness remains somewhat elusive. For example, consider the digital ecology of the living room in Figure 1. It includes various devices (the digital surface, the information appliances, and the things people carry such as smart phones and tablets). While most devices are networked, actually inter-connecting these devices is painful without extensive knowledge, and requires time to configure and debug. Even when connected, performing tasks across devices is tedious, often requiring complex navigations across interfaces. In practice, this means that – from a person's perspective – the vast majority of devices are blind to the presence of other devices. What makes this even more problematic is that these devices are also blind to the non-computational aspects of the room – the people, other non-digital objects, the room's semi-fixed and fixed features – all which may affect their intended use.

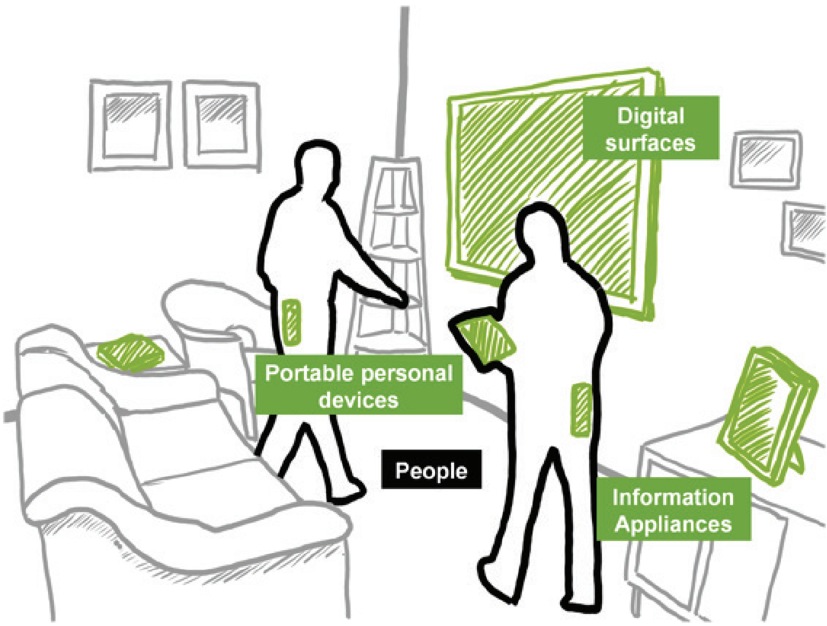

Figure 1: A typical Ubicomp ecology, including a mix of people, digital surfaces, portable personal devices, and information appliances (Ballendat, Marquardt and Greenberg 2010).

This is where we (along with a few others) saw the role of proxemics theory (e.g., see also Vogel and Balakrishnan 2001; Ju et. al. 2008). The main idea is: just as people expect increasing engagement and intimacy as they approach others (as suggested by proxemics theory), so should they naturally expect increasing connectivity and interaction possibilities as they approach devices, and as they bring their devices in close proximity to each other and to other things in the ecology.

Stage 2. Translating Proxemics Theory to Technological Design

Proxemics theory relies both on people's ability to sense their environment and others within it, and on people to interpret what they see to adjust their social behaviors. Technology, of course, is much more limited.

We thus had to translate proxemics theory into a form that we could use as our design foundations. The first question was "what should the system be able to sense?" where our constraints were that these sensing capabilities could be something we could operationalize and implement, that is, as proximity measures in the form of variables returned by the system. Our own notion of proxemic dimensions for Ubicomp are characterized in Figure 2 and explained below, where we consider proxemics measures between entities (entities can be people, devices, and/or physical features in the environment). As we will see, each of these dimensions can also vary by fidelity and whether they return discrete or continuous values.

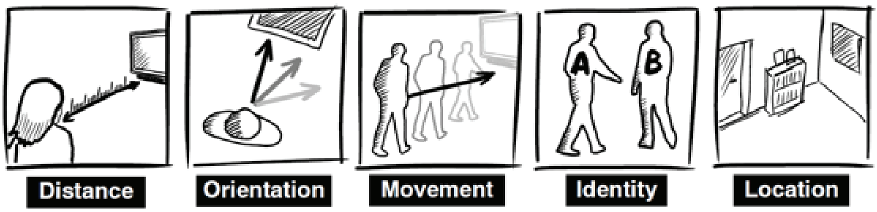

Figure 2. Five proxemics dimensions for Ubicomp (Greenberg et. al. 2011).

Distance between entities is a fundamental notion in proxemics theory. We normally think of distance as a continuous measure, such as a value returned between 0 - 6 feet. However, distance can also be discrete, for example, a measure of what zone an entity is in with respect to another entity. In the simplest case, distance can be considered as a binary measure, e.g., one entity is either near or not near to another entity.

Orientation between entities is also fundamental in proxemics theory. For example, the 'social distance' between two people facing towards vs. away from one another is extremely different, even though the physical distance is identical. Orientation thus captures nuances not provided by distance alone. It too can be continuous (e.g., the pitch/roll/yaw angle of one object relative to another), or discrete (e.g., facing towards, somewhat towards, or away from the other object). Of course, orientation only makes sense if an entity has a 'front face' to it.

Movement between entities captures the distance and orientation of an entity over time, where different actions can be taken depending on (for example) the speed of motion, and/or whether one entity is moving and turning towards vs. away from another entity. People naturally consider movement as part of the social distance dynamics of proxemics. Technology must be informed about that movement as well.

Identity uniquely describes the entity. While proxemic theory is applied to people, we expected we would apply it to a broad range of technical devices as well as physical artifacts within the environment. Thus design requires some degree of entity identification. Identity can range from a detailed measure including exact identity and attributes of that entity, to a less detailed measure such as an entity's type, to a minimal measure that simply discriminates one entity from another.

Location context describes the physical context that the entities reside in. People naturally consider location as part of their behaviors, for example, how a couple adjusts their distancing in a family room versus in a public setting such as a store. Yet technology is blind to context unless explicitly informed. Location measures can also capture contextual aspects, such as when an entity crosses a threshold (a fixed feature) marking its presence in a room. Location is important, as the meaning applied to the four other inter-entity measures may depend on the location's context.

While we will not delve into it here, our choice of these particular measures were heavily influenced by our thinking about how proxemics theory could address known challenges in designing Ubicomp systems (Marquardt and Greenberg, 2012). For example, one of the Ubicomp challenges we considered was establishing connections between devices as a consequence of proximity (e.g., a mobile phone and a surface). A simple thought exercise reveals the importance of distance, movement, and orientation to avoid accidental connections: i.e., a person's intension to connect the phone to the surface would be triggered by pointing and moving the phone towards the surface until a particular close distance is reached. Identity is, of course, important for security reasons. Location context is similarly important, for it may allow some people to connect (e.g., an employee using a board room, where the connection re-establishes particular information) but disallows others (e.g., an unescorted visitor).

Stage 3. Quick and Dirty Prototyping

We then developed many quick and dirty prototypes, often using some quite simple technologies. Various examples are described in detail in Marquardt and Greenberg 2015 and in Greenberg et. al. 2011, as well as in many individual research publications. For example, one of our first prototypes used simple off the shelf range finders as a way to control connection and privacy in an always-on media space (Greenberg and Kuzuoka, 1999). The idea was that people would be able to see and hear each other in increasing fidelity as a function of both actor's proximity to their displays.

Our second prototype realized a cartoon actor (a face) on a large surface. Using a few fairly simple proxemics rules, the face would react to people's distance, movement and orientation. For example, its eyes would track the moving person. The face would verbally great an approaching person, smile as they came closer, frown and get annoyed if they were too close, be sad when they turned away, and so on (Diaz-Marino and Greenberg 2010). We found this application interesting because (a) people with no technical background immediately understood the system's behaviors in terms of how it reacted to their distance, movements, and orientation, and (b) this was in spite of the system following only a few simple proximity-based rules to drive its behavior (it had no artificial intelligence). For our third prototype, we wanted to see what we could do if we added proximity awareness to a traditional presentation tool running on a vertical surface, where the speaker would not have to use a second display or a keyboard. We focused on two specific capabilities: we wanted to make it easier for a speaker to access their speaker notes, and we wanted to make it easier for a speaker to control their slides. For example, when the speaker faced the audience, slides were presented in full. However, if the speaker faced the screen and stood close to one of its sides, speaker notes along with a few navigation controls appeared in the corner closest to the speaker. If the speaker shielded the display from the audience by standing near the middle of the surface, a scrollable deck of slide thumbnails appear, allowing the speaker to rapidly switch to any slide.

These and other applications influenced our thinking about proxemics. They helped solidified our translation of proxemic theory into operational variables (as discussed in the previous stage), and they also influenced our design of the first version of our proximity toolkit (the following stage, discussed next).

Stage 4. Building a Toolkit for Rapid Development

Building proxemics-aware applications are challenging. While rough measures of distance can be captured by range finders, their accuracy proved less than ideal. Capturing other parameters, such as orientation and directional movement proved even more difficult. Programming raw input streams from these sensors was tedious. Simply put, the technical effort of building these systems meant that we spent more time programming the underlying plumbing, which came at the expense of exploring the design space of proxemics.

We turned to a new goal, where we wanted to simplify the exploration of interaction techniques by supplying fine-grained proxemic information between people, portable devices, large interactive surfaces, and other non-digital objects in a room-sized environment. Our solution was the Proximity Toolkit (Marquardt, Diaz-Marino, Boring and Greenberg 2013). The toolkit offered three key features. First, it facilitated rapid prototyping of proxemic-aware systems by supplying developers with the orientation, distance, motion, identity, and location information between entities, all accessible via simple-to-program callbacks. Second, it included various tools, such as a visual monitoring tool, that allows developers to visually observe, record and explore proxemic relationships in 3D space, which helped them understand the data being generated by the toolkit before any coding was actually done. Third, its flexible architecture separated sensing hardware from the proxemic data model derived from these sensors, which meant that a variety of sensing technologies can be substituted or combined to derive proxemic information. We initially based our hardware infrastructure on the Vicon Motion Capture system, where the system would return millimeter-accurate data about an entities position in 3D space. However, later versions incorporated other sensing systems, such as the lower-cost Optitrack motion capture system, and the consumer-affordable Microsoft Kinect depth-sensing camera.

Callbacks follow standard programming conventions to track events. For example, consider a simple scenario where a programmer wanted to display information only if a person was facing the display. The callback would be something like:

```
void OnDirectionUpdated (ProximtySpace space, DirectionEventArgs args)
{
  if (args.ATowardsB)
      //Person is facing the display, show content]
  else
      //Hide content
}
```

We developed several versions of the toolkit over a few years. While it took considerable time and effort to do so, the result was well worth it. Programmers with only a brief introduction to the toolkit were able to create proxemics-aware applications almost immediately. More importantly, complex applications could be built, where programmers could concentrate and iterate over the design of particular proxemics-aware systems.

Stage 5. Robust Design and Development.
By this stage, we had developed a solid understanding of proxemics and how it could be applied to the design of systems supporting proxemics interactions. We also had a toolkit that let us actually build, maintain, and iterate through fairly complex proxemics-aware systems. A few examples illustrate what we could do.

Proxemic Media Player is a media player that reacts to the proximity of one or more people in a room (Ballendat, Marquardt and Greenberg 2010). Figure 3 illustrates only a few of its functions. At distance (a), the person enters the room. The media player recognizes both the person's identity and entrance, activates the display, shows a short animation, and then displays four large video preview thumbnails held in that person's personal media collection at a size suitable for distance viewing. At distance (b) the person is moving closer to the display. The display responds by showing an increasing number of his videos by continually shrinking the video preview thumbnails and titles to fit. At distance (c), the person is very close and he can select a video to watch by directly touching its thumbnail, which shows him more about the selected video: a preview that can be played and paused, with detailed title, authors, description and release date. The text is small, but quite readable at this close distance. Finally, at distance (d) the person moves away from the screen to sit on the couch. The system responds by expanding the currently selected video to play in full screen view. When seated at the couch, the person can also point his mobile phone towards the display. The phone is recognized as a pointing device, which in turn can be used to control the media player. If a second person enters the room, the video shrinks slightly to expose the title of the video being played. If that second person then approaches the screen, a description of the video is revealed. When all people leave the room, the video playback stops.

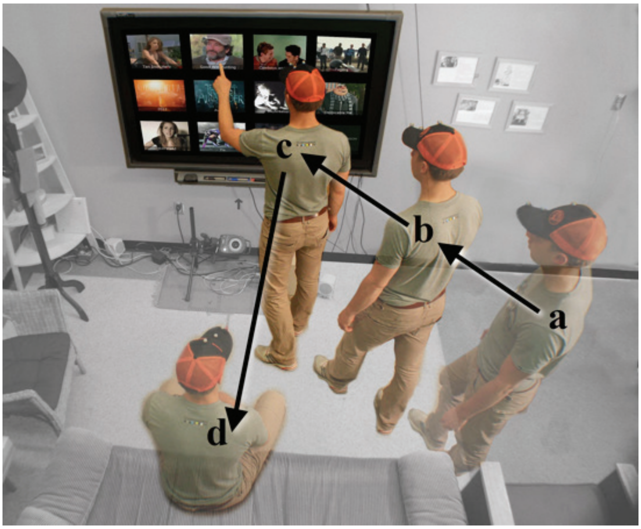

Figure 3. Proxemics Media Player. The position of a person in the room is shown at the top, where letters correspond with what the surface is displaying at those distances (Ballendat, Marquardt and Greenberg 2010).

The Gradual Engagement design pattern is a generalizable interaction technique that describes what we believe is a successful way to exploit proximity (Marquardt, Ballendat et al., 2012). The general idea is that we can design devices and interfaces that interpret decreasing distance and increasing mutual orientation between a person and a device within a bounded space as an indication of a person's gradually increasing interest in interacting with that device. The generalized gradual engagement design pattern includes three key phases:

- Phase 1: background information supplied by the system provides awareness to the person about opportunities of potential interest when viewed at a distance;

- Phase 2: the person can gradually act on particular opportunities by viewing and/or exploring its information in more detail simply by approaching it; and
- Phase 3: the person can ultimately engage in action if so desired.

This pattern is directly inspired by the proxemic theory mentioned earlier, and characterises what we thought was the 'best' of how we, and others previously, apply proxemics to Ubicomp design. The intention of this gradual engagement pattern is to characterise how we can facilitate interactions between a person or multiple people and the devices surrounding them by leveraging fine-grained proxemic measurements (e.g., distance, orientation, identity) between all entities. As a design pattern, it helps unifying prior work in Proxemic Interactions, synthesizing essential, generalizable interaction strategies, and providing a common vocabulary for discussing design solutions.

We noticed that many of our early designs incorporated the idea of gradual engagement, for example, the media player, where details of the videos available are revealed as a person approaches the surface, and where interaction techniques are tuned to allow finer interactions (using touch) when the person enters the intimate zone. Furthermore, the Gradual Engagement design pattern also informs and inspires other possible designs, and allows for variations of the pattern applied to different domains. The remaining examples illustrate this broad application of the pattern.

Gradual Engagement Pattern for Cross-Device Information Exchange. In this first example, we applied the design pattern to mediate device-to-device operations. In particular, we refined the gradual engagement pattern to ease the information transfer task, where the refined pattern suggests how devices can gradually engage the user by disclosing connectivity and information exchange capabilities as a function of inter-device proximity. That is, as people move and orient their personal device towards other surrounding devices, the interface progressively moves through three stages affording gradual engagement.

1. Awareness of device presence and connectivity is provided, so that a person can understand what other devices are present and whether they can connect with one's own personal device. We leverage knowledge about proxemic relationships between devices to determine when devices connect and how they notify a person about their presence and established connections.

2. Reveal of exchangeable content is provided, so that people know what of their content can be accessed on other devices for information transfer. At this stage, a fundamental technique is progressively revealing a device's available digital content as a function of proximity.

3. Interaction methods for transferring digital content between devices, tuned to particular proxemic relationships and device capabilities, are provided via various strategies.

Each method is tailored to fit naturally within particular situations and contexts. As one part of this pattern, Figure 4 demonstrates the proximity-dependent progressive reveal of digital content stored on personal devices when collaboratively interacting with a large shared interactive whiteboard.

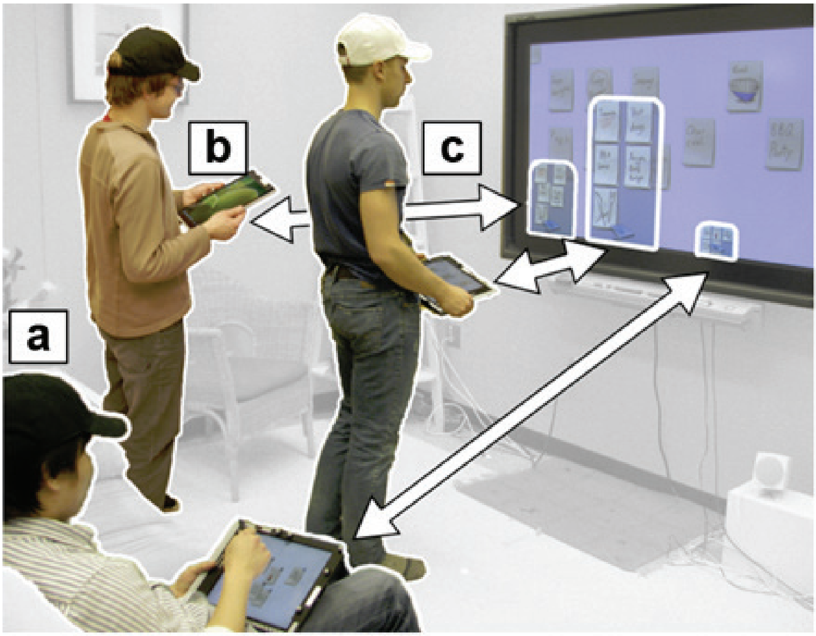

Figure 4. Proximity-dependent progressive reveal of personal device data of multiple users at different distances to the display: (a) minimal awareness of a person sitting further away, (b) larger, visible content of a person moving closer, and (c) large awareness icons of person standing in front of the display (Marquardt, Ballendat et al., 2012).

Gradual Engagement with Proxemic-Aware Advertisements. A second application of the design pattern was the Proxemic Peddler that explores how future advertisement displays might try to grab and keep a passer-by's attention (Wang, Boring and Greenberg, 2012). A digital advertisement board – in this case a book-selling display – reacts to the presence, distance, identity, orientation, and movements of a nearby person. The key is to do so in a non-aggressive and non-annoying manner that finds a balance between the advertiser's interest and the passer-by's interest. When no-one appears within its range, it rapidly animates a book list at the bottom, where its motion is an attempt to attract the attention of a passer-by. The animation slows as soon as it detects a passer-by looking towards it (which makes the book list readable and far calmer), as illustrated in Figure 5, upper left. The gradual engagement pattern is then applied, where additional personalised details about preferred books are displayed as the person approaches the display (Figure 5, upper right). If the person momentary looks away, subtle cues are used to try to re-attract them, such as a slight shaking of the product icon (Figure 5, lower left). If it looks as if the person is about to leave, it tries to regain their interest by showing different products (Figure 5, lower right). In all cases, it gives up gracefully if it looks like the person is really not interested.

Figure 5. Proxemic Peddler (Miaosen Wang).

Proxemic-based remote controls leverage Proxemic Interactions in order to mediate the control of appliances in a person's Ubicomp environment (Ledo, Greenberg, Marquardt, Boring 2015). Using a mobile device (e.g., phone or tablet, Figure 6 left) as a personal control device, a person can initially point around the room in order to scan which devices are available. Items coming into view on the display are the ones generally in front of the device. The person can then gradually increase the control of a particular appliance simply by moving closer to it. More details about the appliance's current status and activity are shown on the screen, and the interface reveals further control options to take action. For example, in Figure 6 (right) the progressively revealed stages of a temperature control interface to a physical thermostat are shown, from small icons on the left progressing to detailed graph views of recent activity on the right. In summary, these proxemic-aware controls are an alternate yet complementary way to interact with appliances in people's environments via a mobile device. Through spatial interactions, people are able to discover and select interactive appliances and then progressively view its status and controls as a function of physical proximity. This allows for situated interaction that balances simple and flexible controls, while seamlessly transitioning between different control interfaces.

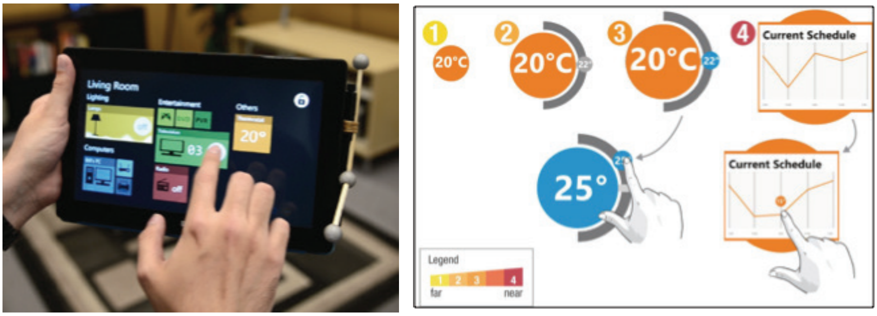

Figure 6. (Left) Proxemic-aware remote controls: remote control interface on a tablet computer; (Right) thermostat interface, showing a series of progressively revealed interaction controls on the remote control's screen (Ledo, Greenberg, Marquardt, Boring 2015).

Summary

This chapter described the five interleaved stages of a research pattern, where its basic premise is to use social science theory to motivate design. Using proxemics theory as a case study, we illustrated how we applied this pattern to co-develop the design notion of Proxemic Interactions along with a toolkit and a broad set of prototype systems.

We are sometimes asked if our work is driven by theory, or whether it is just inspired by theory. The answer is perhaps a bit of both. With theory-driven research, we rely on that theory to frame the behavior of our system-as-actor, where the behavior should correspond (at least to a reasonable extent) to that theory. Similarly, we rely on the theory and its nuances to explain and predict how people will likely respond to our design ideas. However, we do not blindly follow the theory, as we recognize that technology cannot simply be substituted in place of one of the humans. We allow ourselves to go beyond the theory. That is, we use the theory as a starting point to help inspire designs, but are not concerned when our interaction ideas stretch that theory or go beyond what the theory says. We are also open to creating new 'theories' that incorporate technology as one of the actors. For example, our design pattern of gradual engagement is a theoretical variation of proxemics. As such, the gradual engagement pattern offers an interaction technique that can be applied to many technology settings, and that incorporates what we believe are good technological behaviors that are easily understood and beneficial to people.

Design creativity does not have to occur in a vacuum. This chapter offers social science theory a contributor to both the initial design spark and for shaping design alternatives over the course of the design process. Our book "Proxemic Interactions: From Theory to Practice" (Marquardt & Greenberg, 2015) adds considerable detail to what is provided here.