Range: Exploring Implicit Interaction through Electronic Whiteboard Design

Wendy Ju, Brian A. Lee, & Scott R. Klemmer
Stanford University
353 Serra Mall, Stanford CA 94305 USA
{wendyju, balee, srk} @stanford.edu

ABSTRACT
An important challenge in designing ubiquitous computing experiences is negotiating transitions between explicit and implicit interaction, such as how and when to provide users with notifications. While the paradigm of implicit interaction has important benefits, it is also susceptible to difficulties with hidden modes, unexpected action, and misunderstood intent. To address these issues, this work presents a framework for *implicit interaction* and applies it to the design of an interactive whiteboard application called Range. Range is a public interactive whiteboard designed to support co-located, ad-hoc meetings. It employs proximity sensing capability to proactively transition between display and authoring modes, to clear space for writing, and to cluster ink strokes. We show how the implicit interaction techniques of *user reflection* (how systems indicate to users what they perceive or infer), *system demonstration* (how systems indicate what they are doing), and *override* (how users can interrupt or stop a proactive system action) can prevent, mitigate, and correct errors in the whiteboard's proactive behaviors. These techniques can be generalized to improve the designs of a wide array of ubiquitous computing experiences.

Author Keywords
Implicit interaction, foreground/background, proactive, proxemics, ubiquitous computing, whiteboards,

ACM Classification Keywords
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*input devices and strategies, interaction styles*.

CSCW'08, November 8–12, 2008, San Diego, California, USA.

INTRODUCTION
One of the defining traits of ubiquitous computing is the pursuit of invisibility. Different camps of interface researchers and designers have taken different tacks towards this elusive goal. This is evidenced by the amazing diversity of ubiquitous computing genres which cite Mark Weiser's "Computer for the 21st Century" [32] as a genesis—ambient displays, tangible user interfaces, context-aware computing, attention-sensitive interfaces, just to name a few. In light of this great variety of approaches towards invisibility, it is useful to keep in mind that invisibility, as championed by Weiser, is not so much about staying beneath notice as enabling seamless accomplishment of task.

In their paper, "Making Sense of Sensing Systems: Five Questions for Designers and Researchers," Bellotti *et al*. point out that ubiquitous computing systems are particularly susceptible to problems of unintended actions, undesirable results, and difficulty detecting or correcting mistakes [1]. This occurs because of the high potential for miscommunication when the interaction between the computing system and the user occurs beneath the user's notice or without the user's initiative. Since invisibility is about enabling seamless accomplishment of desired tasks rather than evading notice, we propose that it is important to understand how to design transitions between explicit and implicit interaction, so that users can make requests, anticipate actions, and make corrections in a robust manner even in situations where they have limited attentional, cognitive, or physical bandwidth for interaction.

The goal of this paper is to explore the range of ways that designers can establish shared understanding between user and system without using keyboard, mouse, or stylus for input, and without using dialog boxes for output. To accomplish this task, we present a framework for implicit interaction, as a well as an implementation of a ubicomp whiteboard application, from which we extrapolate general purpose implicit interaction techniques. It is our hope that this framework and illustration will help to add implicit interaction design to the range and repertoire of ubicomp interaction designers.

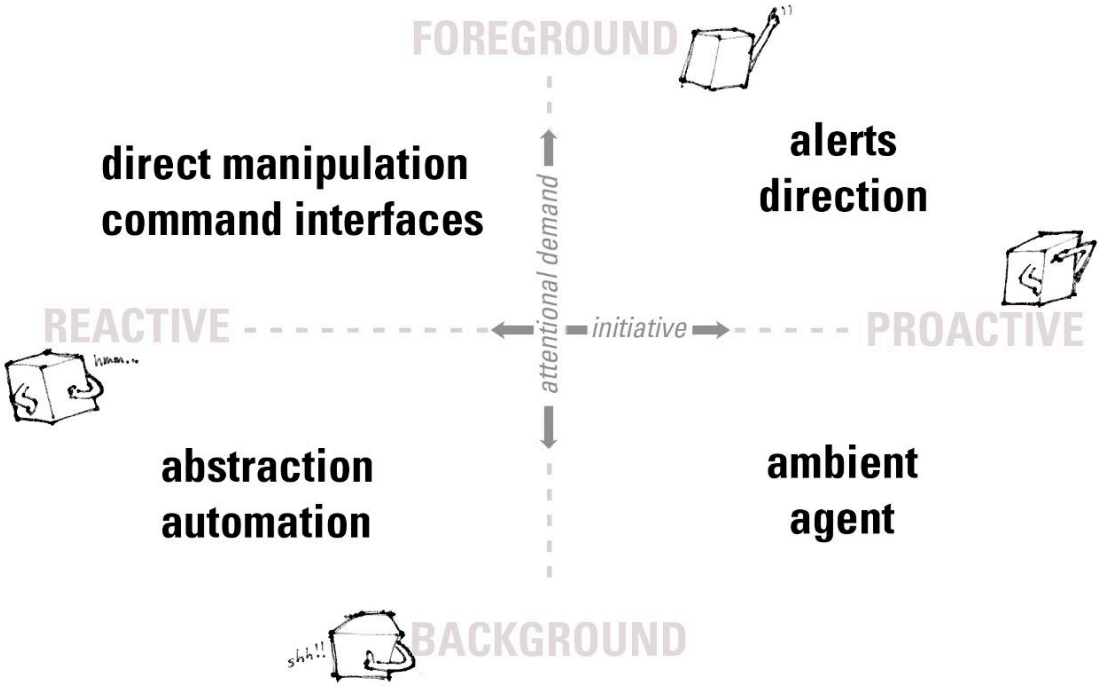

Figure 1. The Implicit Interaction Framework is based on two axes: the level of attentional demand the system places on the user and the balance of initiative taken by the system on behalf of the user. This framework provides a domain-independent characterization of an interaction's implicitness.

IMPLICIT INTERACTION FRAMEWORK

Implicit interactions are an inevitable part of "smart" products, whose actions contain some degree of agency of activity that occurs without the explicit behest or awareness of the user. Implicit interactions enable communication and action without explicit input or output. One way that an interaction can be implicit is if the exchange occurs outside the attentional foreground of the user. This occurs in traditional computing–when the computer auto-saves your files, or filters your spam e-mail, for instance–as well as in ubiquitous computing interaction. The other way that an interaction can be made implicit is if the exchange is initiated by the computer system rather than by the user–if the computer alerts you to new mail, say, or when it displays a screen saver. It may seem counter-intuitive that something that grabs attention could be implicit, but the key factor is that the interaction is based on an implied demand for information or action, not an explicit one.

The implicit interaction framework (see Figure 1) divides the space of possible interactions along the axes of attentional demand and initiative [15]. Attentional demand is the degree of cognitive and perceptual load imposed on user by the interactive system [23]. *Foreground interactions* require a greater degree of focus, concentration and consciousness, and are exclusive of other focal targets, while *background interactions* are peripheral, have less demand and can occur in parallel with other interactions [34]. Initiative is an indicator of how much presumption the interactive system uses in the interaction. The framework presumes the perspective of system designers, so interactions that are initiated and driven by the user explicitly are called *reactive interactions*, while interactions initiated by the system based on inferred desire or demand are *proactive interactions* [29]. By characterizing interactions in this way, we are able to generalize about the capabilities and features of whole classes of interactions in a domain-independent fashion.

Let us examine these two dimensional variables in greater depth:

Attentional Demand

Attentional demand does not correspond easily with any particular metric, in part because attention is very complex [3]. Any comprehensive definition needs to account not only for the *load* on the resource of cognition [18], but also for *spatialization* (whether something is in the center or the periphery of one's notice) [34], *breadth* (whether attention is focused on a single stimulus or many), and *gestalt* (whether attention is devoted to the abstracted whole or the individual parts) [30]. The other challenge that researchers have identified is that attention—by its very nature—can be challenging to evaluate directly [23].

Interaction designers commonly manipulate attentional demand by adjusting the perceptual prominence of objects, often implicitly, through visual organization techniques such as contrast, hierarchy, and weight [35]. Demand may also be choreographed through more dynamic means, such as pointing (e.g. calling attention to an object by gesturing

at it) or placing (e.g. calling attention to an object through its prominent placement) [7]. Still another way to affect the degree of attention demanded is through abstraction and chunking, whereby small interactions are combined into a larger whole [4].

Initiative

Initiative is salient in situations where actors are working together to accomplish a task, and can be highly contextual. If a word processing program saves your document because you command it to, it is clearly reactive. If it auto-saves your document because you have set it to do so every 10 minutes, it may do so out of your attentional center, but it is still responding to your explicit command. However, if the same program saves your program because it feels that a lot of changes have been made, it is more proactive; it is operating in a realm of greater presumption with respect to the needs and desires of the user.

Designers can manipulate the proactivity and reactivity of a designed interaction by dictating the order of actions—does the system act first, or wait for the user to act?—as well as by choosing the degree of initiative—does the system act, offer to act, ask if it should act, or merely indicate that it can act? Designers also affect the degree of initiative when they gather more data to ensure the certainty of the need for an action or when they design in features to mitigate the potential cost of error for the action. Even in the reactive realm, the degree of initiative can vary based on the amount that the user needs to maintain ongoing control and oversight of an action in progress.

Types of Interactions

The following are descriptions of interactions and illustrating examples for each quadrant:

Reactive/foreground

Interactions take place explicitly and at the user's command. Users are given explicit and detailed oversight over actions and feedback on results. Such interactions are appropriate when the interaction is the primary task and is controlled by a knowledgeable user. Normal GUI interaction falls in this quadrant.

Reactive/background

Interactions occur in response to user actions or external stimuli, but feedback is generalized or hidden from the user (abstraction). Such interactions can spare the user from the nitty-gritty details of a task or help perform routine tasks automatically with little or no user oversight (automation). The "auto-save" on a typical word-processing program, which is based on time elapsed, exemplifies this type of interaction.

Proactive/foreground

Interaction takes place in the attentional foreground, but involves greater urgency on the part of the system. The system may provide unsolicited information (alerts) or guide the interaction by instructing the user what to do (direction). These interactions are typical in reminder and tutorial scenarios. The "You've got mail" sound and bouncing icon in typical e-mail program are examples of proactive/foreground interactions.

Proactive/background

The system anticipates what to do and performs actions with low oversight or input. These interactions are usually used for tasks where the cost of error is low: for instance, pre-fetching data, or modeling preferences. They may also be employed to address critical tasks that the user is somehow unable to perform, like alerting the police when someone is intruding into one's home.

Implicit Interaction Patterns

While it is possible to speak of implicitness or explicitness as genres of interaction, the key value of the implicit interaction framework is its ability to illuminate the dynamic transitions between the quadrants in successful interaction sequences. By framing effective interactions in terms of the dynamics of attentional demand and initiative, the framework illuminates patterns of social interaction, which makes it distinct from frameworks which emphasize patterns of domain-specific solutions, such as [5], or context-specific routines, such as [30]. Thus, interaction designers may more easily recognize and reason how and why existing implicit interactions function, and leverage that understanding in designing implicit interactions for novel applications where domain precedents and conventions may not exist.

To explore the design of implicit interactions, we have applied the implicit interaction framework to the design of new features for an interactive whiteboard. In the following sections, we will discuss our selection of interactive whiteboards as the domain for our exploration, review related work on implicit interactions and whiteboards that informed our framework and interaction design, outline the specific design our electronic whiteboard system, Range, and discuss the implicit interaction techniques illustrated by our implementation.

INTERACTIVE WHITEBOARDS AS A TESTING GROUND FOR IMPLICIT INTERACTION

The ephemeral nature of whiteboard ink allows users to share ideas quickly—and just as quickly, to amend those ideas. The improvisational quality of whiteboard use is a good match for the provisional ideas that are generated in informal design meetings, when people are more concerned with entertaining possibilities than communicating fact. The ubiquity of whiteboards in dedicated design spaces (such as war rooms, and project rooms) and informal meeting spaces (such as offices, break rooms, and hallways) is a testimonial to the utility of the whiteboard to designers everywhere.

The utility and ubiquity of whiteboards makes them an appealing platform for computational enhancement. However, the attractive aspects of whiteboards are

inextricably linked to the factors that also make them challenging to augment. The shared, public nature of whiteboards means that the interface must succeed for walk-up use, and the focus on quickly sharing ideas means that any services provided must have a low threshold to entry and minimal attentional overhead [27].

The issues associated with whiteboards are like those of many ubiquitous computing situations: interactions are often transient and needed on-demand; and the users are often distracted and untrained. We have introduced this whiteboard as an implementation that helps manifest the opportunities for and challenges with implicit design in ubiquitous computing.

RELATED WORK

This paper draws on related work in three areas: the framing of interaction styles, workplace studies of whiteboard usage, and the design of electronic whiteboards.

Design Frameworks for Implicit Interaction

The framework laid out in this paper builds on Buxton's foreground/background model [2]; in it, Buxton distinguishes foreground interactions—to paraphrase, intentional activities that take place in the fore of human consciousness—from background interactions, for example, lights that automatically turn on when you enter a room—which take place in the periphery of consciousness [34]. This model identifies the same attention and initiative used in our framework, but assumes the two are inherently linked. Actions initiated by the user are assumed always to be taken with intent; actions taken by the system are assumed to take place in the periphery. Our framework amends Buxton's framework by decoupling attention and initiative into separate axes. Buxton's foreground corresponds to our reactive/foreground quadrant, and his background corresponds to our proactive/background.

The implicit interaction framework bears some commonality with Pederson's model for tacit interaction [24]. Pedersen's framework models tasks based on the degree of attention and focus required and the degree of intentionality in action. However, the tacit interaction framework describes interactions only in terms of the user's degree of intention. No distinction is made between situations where users don't have to think and plan because the users have developed tacit knowledge of how to operate a task, and situations where the users don't have to think and plan because the system is acting proactively on their behalf. In a well-designed interaction, the user may be unaware of the system, but this is the effect of successful implicit interactions, and not the cause. Our framework aims to illustrate specific methods of achieving seamless interactions.

Horvitz *et al.* [13] present a related model for notification displays. This framework uses an economic model of user attention, and determines the expected utility of presenting users with notifications, based on the level of attentional cost to the user and the expected value of the information. This model traverses the same territory as the right side of our framework, ranging from proactive/foreground to proactive/background. Its use of uncertainty as a measure of proactivity guided our framework's formulation of initiative. This model is ideally suited to help computers make dynamic determinations about the right way to deliver a piece of information. It provides less guidance, however, to the interaction designers developing the different methods the computer might eventually chose from.

Workplace Studies of Whiteboard Usage

The Flatland whiteboard interface [23] was based on informal observations of whiteboard use in office settings. Researchers observed that office use of whiteboards was characterized by thinking and pre-production tasks, everyday content (such as task lists, sketches, and reminders), clusters of content (both persistent and short-lived), and a transitioning between semi-public to personal use. Our design of Range builds on the observations that Flatland is based on. It includes features to support a range of use from display to whiteboard, freeing up space for drawing, and clustering strokes of ink. The major departure in our explorations is the use of distance sensing as input for these features, and the avoidance of meta-strokes or other explicit techniques.

Longitudinal studies of student engineering design teams working on multi-month projects by Ju *et al.* [14] found that engineers engaged in informal meetings would cycle between phases of drawing and analysis; these changes corresponded with changes in their physical proximity to the whiteboard. Users would stand close to the board when they were writing, further back when discussing written artifacts in detail, or further back still when engaging in meta-discussion. They also found that input was initially free-form, but that meeting participants would often close their meetings by performing post-facto structuring on previously generated sketches, drawing borders, lines, and arrows to explicitly group or relate elements on the board.

Our observations of whiteboards, based on photos taken around campus in several departments, indicate that sketches on the board can generally be categorized as either "read-only" or "write-only." What we called "read-only" were messages that were meant to persist, and changed infrequently: phone numbers of colleagues, lists of upcoming deadlines. Sketches that were "write-only" were usually generated in informal meetings, and were infrequently referenced after their initial creation. Regardless of field, people implicitly placed information that is meant to be static or saved along the edges of the board, saving the center of the board for temporary and speculative work. This finding validates location of information on the board as a crucial context variable.

Figure 2. Physical setup of Range (left), with diagram of interaction zones (right).

Design of Electronic and Augmented Whiteboards

Electronic whiteboards emerged out of the ubiquitous computing research at PARC, and their goal of computing by the inch, foot, and yard [33]. PARC's LiveBoard [8] was a rear-projected electronic whiteboard that afforded pen-based input through infrared-emitting styli. Tivoli [25], the LiveBoard's whiteboard application, introduced a set of interaction techniques for creating and manipulating ink-based documents, and supported input from multiple pens simultaneously. Ink strokes were stored as grouped vector objects, and the system introduced gestures for the selection, grouping, and manipulation of ink content.

Subsequent research [21, 22] explored the use of implicit structure in the spatial layout and proximity of users' inkmarks. In grappling with whether such implicit structures should be exploited by the electronic whiteboard as input, or if input should be wholly freeform, the PARC researchers introduced the first pen-based interface to decouple *recognition* (having the system create an internal hypothesis of the user's intended structure) from *transformation* (having the system in turn modify the representation of the user's data based on its belief about the structure). These ideas were extended upon in SILK [17] and subsequent informal user interfaces, e.g., [19] [16]. This selective and timed presentation of what the system infers is used in our design of Range.

Recent work on electronic whiteboards has focused on incorporating aspects of the user's physical context in whiteboard use into the interaction. Research on using paper and digital artifacts with an electronic whiteboard [16], on using pen-based command techniques for high-resolution displays [10] and on physical gestures [28] and tokens [20] for specifying behaviors begins to realize Weiser's vision of computation that is embedded into the fabric of everyday life. Current work in ambient interfaces is also exploring the understanding of the user's physical context as an implicit input in the domain of large interactive public displays. Both Prante *et al.*'s Hello.Wall [26] and Vogel & Balakrishnan's interactive Ambient Public Displays [31] stand out for explicitly noting the proxemic relationship between the physical distance between multiple users and the display, and for applying that information to modify the contents of the display accordingly. Our whiteboard design draws on similar proxemic relationships between users and whiteboards, but the implicit meaning of the being close or far from each board differs because whiteboards are intrinsically meant for writing as well as display.

This paper offers two contributions beyond this work in electronic whiteboard interactions. The first is that it provides a richer framework for describing and designing implicit interactions; the second is that it is oriented towards broadening the range of interactive technique rather than the enriching the pool of whiteboard features.

THE RANGE WHITEBOARD

To illustrate how implicit interaction techniques can be used to prevent, mitigate and correct the problems of proactivity in the area of whiteboard interaction, we designed an interactive whiteboard named Range, which uses infrared distance sensors to subtly and proactively interact with informal meeting participants.

Implementation

Range was implemented using a combination of pre-existing hardware and software tools and technology.

Platform

The Range whiteboard prototype employs a rear-projection SMART Board containing an SXGA+ resolution projector (1400x1050) and a Windows XP PC. Four SHARP GP2Y0A 150 cm analog distance sensors were mounted to the front of the board, and connect to the PC over USB via

the d.tools hardware and libraries [12]. The software component of Range was written in C# using the Microsoft Tablet PC SDK and the SMART Board SDK.

Physical Interaction Design

The region in front of the board is divided into four zones, which we called intimate, personal, social, public in reference to proxemics pioneer Edward T. Hall's distance zones [11]. We defined the intimate zone to be the region in which users stand to write at the board, testing with multiple users to increase the robustness of the zone definitions. The personal zone was set further back, at a distance (>15 inches back) where users were not "at" the board, but could easily reach the board for pointing and text manipulation. The social zone (>25 inches back) was out of touching distance from the board but in easy viewing distance of the board. The public zone comprises the distance beyond the social (> 40" back).

The operational zone was based on the user closest to the board; we found that this is usually the person with the pen and thus the person "driving" the interaction at the whiteboard.

Operation

The SMART Board uses a pen tray with four colored styli and an eraser. Strokes made with the styli make ink strokes of the corresponding color on the board, and strokes made with the eraser remove marks intersected by the erase stroke. Input on the capacitive board is presumed to be made by the users' fingers if all the styli are in the tray; such finger input is used to select and move ink strokes and clusters.

We modified the SMART Board operation so that inputs issued when the user is in the personal zone are read as select and move operations even if the pen is out of the tray; this seems more natural to users and lessens the instances of erroneous input.

Features

We implemented three features in Range that demonstrate implicit interaction techniques: an automatic transition from ambient display to drawing space, automatic space clearing, and automatic ink stroke clustering.

Transition from Display to Drawing Surface

When users are not engaged with Range, the whiteboard switches to ambient display mode, overlaying the existing whiteboard contents with a transparent blue backdrop and a stream of digital images of interest to users. Our implementation uses snapshots of previous whiteboard states and other photos of interest from an online photo-sharing site to improve project awareness.

As a user approaches a Range whiteboard in screensaver mode, the backdrop fades and the displayed screensaver content floats off to one side, allowing the user to re-engage the whiteboard contents beneath. If the user touches the departing screensaver content, it stops and becomes selected so that the user may move it to some place on the whiteboard of his or her choosing. We found this "floating" to be important because it helped users to form a model of where ambient images "went to." This metaphor also facilitated correction; users found it "natural" to keep images by grabbing them as they were departing.

Making space

As the designers of Flatland observed, whiteboards are not merely ephemeral objects: people leave drawings or notes on the board in order to provide shared reference for groups [23]. However, a whiteboard full of writing can discourage whiteboard use, as our informal studies found that users are hesitant to erase work. Copying content to another surface takes time, time that may kill a serendipitous, free-flowing conversation.

To address this problem, Range senses "full boards" and moves board contents out to the left and right of the board center when it senses a user approaching, clearing a space so that the user immediately has a blank space in which to write. Data on the edges of the board are not affected during the board-clearing maneuvers.

Clustering Ink Strokes

In order to move text and graphics around while maintaining coherency of the sketches, the underlying system needs to have some conception of the semantic units of whiteboard contents. To achieve this, we have implemented a simple form of stroke clustering, using the stroke's timestamp (time of creation) and location on the board (estimated by its bounding box). As strokes are created, the Range system runs a clustering algorithm in the background: strokes that were either created at the same time (temporal locality) or that are close together on the board (spatial locality) are clustered together automatically.

Users are given feedback about the clusters, by way of dotted light-gray bounding boxes, when they are located in the personal zone. Users manipulate clusters as an atomic unit: selecting one stroke in a cluster selects them all by default, and moving a stroke in a cluster moves the whole cluster. Users may override the automatic clustering by lasso selecting one or more strokes, which puts all of the selected strokes into a new cluster.

IMPLICIT INTERACTION TECHNIQUES

The designs of the three aforementioned features illustrate the implicit interaction techniques of *user reflection*, *system demonstration,* and *override*. These features do not necessarily form an exhaustive set of implicit interaction techniques, but they provide characteristic solutions to interaction problems typical of ubiquitous computing [1].

User reflection

User reflection is how the system indicates what it feels users are doing or would like to have done. User reflection seeks to validate inferred input; this validation corresponds

with what linguists call "recognition and uptake" [6]. Some variations on user reflection are *projections*, which reflect a user's intent, capability, or desire, *feedback,* which reflect a user's actions, and *feed-forward,* which reflect the consequences of a user's actions.

The trajectory of user reflection in the implicit interaction framework's design space is shown for the example of the ambient display feature in Figure 3. It goes from the lower left hand quadrant, where the interactive system is monitoring and responding to user actions in a background fashion, to the upper right hand quadrant, where the interactive system proactively calls attention to its perceptions, actions or potential consequences. Note that the effectiveness of this technique belies the naïve design assumption that implicit interactions are created by staying beneath notice; rather, by implicitly reflecting its perception of the user's actions and intentions, the interactive system can increase the likelihood that it will act in an intuitive or desirable manner.

The range of implicit techniques for reflection can also be wider than that of explicit reflective techniques. For instance, explicit user reflections tend to be intermittent, so they may be interleaved as interacts take turns speaking; implicit user reflections can take place continuously, providing concurrent information about the actions, modes and states they are reflecting. Early spell-checking programs, for example, had to be invoked explicitly, and engaged the user in an explicit dialogue about potentially misspelled words to enable repair. Contemporary spell-check features, however, run continuously in the background, highlighting words that are not in the system's dictionary; users may more easily notice potential errors, but the implicit alert of this interaction is far more seamless than that of earlier explicit spell-check programs.

User reflection is particularly important in the design of ubicomp systems, because the potential input space is so vast that users may have difficulty understanding what sensed action or state triggered downstream actions. When designing user reflections, it is important to map specific features to subsequent acts. It is also useful to perform fieldwork to understand what meaning exists for different reflection displays in different contexts. Grocery store shopping counters [7], for instance, have been designed to confer special meaning to the objects placed on the counter, but the design is not arbitrary. The counters are located so that the placement of objects is in the foreground of both the shopper and the clerk, and so that the counter helps to obscure those objects that are not part of the financial transaction—the bag from the previous store, or your handbag, for instance.

User reflection in Range

User reflection informed the design of nearly all of Range's features. For example, prototype versions of the ambient display mode in Range reflected the physical zone that users were detected in by highlighting the resulting mode.

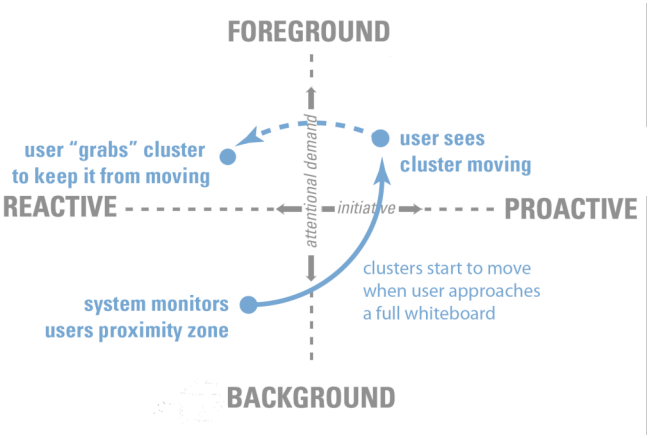

Figure 3. The trajectory of user reflection (solid line) and override (dotted) used in making space.

This lead to confusion, as a stray bystander or blocking object could cause the system to transition unexpectedly. It took collaborating groups a long time to discern what had prompted mode changes. Later versions of the ambient display presented a simple four dot diagram illustrating what zone people were perceived to be in; this reflection made it easier for users to comprehend and repair the situation, thus enabling a more fluid interaction. Similarly, in the making space feature, some objects are deemed to be ephemeral and others to be persistent; inadvertent moving or erasing can be prevented by reflecting the inferred intent by drawing "pins" on read-only clusters.

The outlining of clustered ink strokes is another example of user reflection in the Range application. The ink strokes are implicitly related to one another by their proximity in space and time. This behavior is based on a tacit understanding of proximity and association, but is prone to error. For example, if collaborators draw axes for a graph, and subsequently add points to the graph, graph points may be neither spatially or temporally proximal to the original axes ink strokes, but should still be associated. While a more sophisticated recognition system might be trained to comprehend graph drawings, Range's errors are easily repaired because the clustered ink stroke outlines give users feedback about the systems interpretation of the graph, and enables repair before users try to move the graph and find that the data points do not follow.

Ubiquitous computing systems can take advantage of context-sensitive cues in performing user reflection. In the ink stroke clustering design, for instance, validation occurs when Range outlines the clusters as the user steps back. This moment is opportunistic because it follows the period when the user is actively writing, and should not be interrupted, and usually precedes the period when the clusters of text might be automatically selected and moved.

System demonstration

System demonstration is how the system shows the user what it is doing, or what it is going to do. This differs from

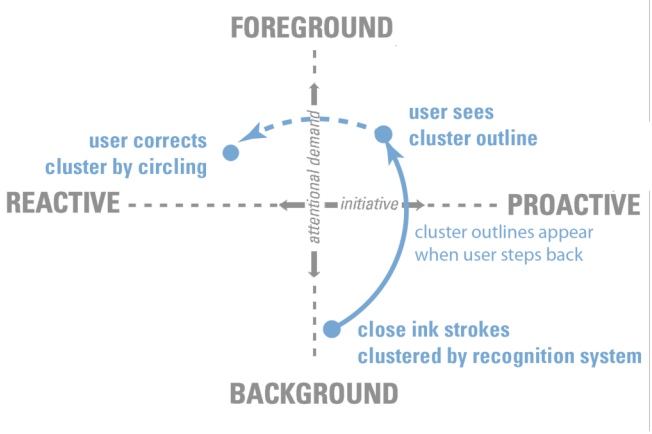

Figure 4. The trajectories of system demonstration (solid line) and override (dotted) used in ink stroke clustering.

the traditional conception of output in that it is not necessarily symbolic, overt, or immediate. When the system "demonstrates," it implicitly asks for the user's attention so that it can make a suggestion or request oversight, thus reducing the likelihood that it will act in an unanticipated or unwanted manner even when its actions are not explicitly prompted. Variations of this technique include *offers*, wherein the system projects potential actions, *demonstration-of-action*, where the system overtly presents on-going action as they take place or immediately afterwards, and *demonstration-of-consequence*, where the system overtly calls attention to outcomes of its actions.

The trajectory of system demonstration is shown in Figure 4. Here, system demonstration of Range's ink-clustering feature is illustrated on the implicit interaction framework. The interaction starts in the lower right hand quadrant, where the system is proactively performing background actions (here, the task of recognizing ink clusters), and moves to the upper right hand quadrant to indicate to users what actions are being taken on their behalf.

The implicit technique of system demonstration is not unique to ubiquitous computing systems; indeed, people use demonstration all the time in their everyday interactions with one another, exaggerating the presentation of their actions–speaking louder, making large showy movements, moving slowly–to enable smoother joint activity [7]. Ubiquitous computing systems may make use of the fact that they have many more potential modes for actuation and demonstration than are available in a traditional computing environment. For instance, as a rule of thumb, small-scale versions of an action (overtly leaning in the direction of the door) are implicitly understood as an offer or request to perform the full-scale action (leaving). However, the design of system demonstrations requires testing with actual users to rule out false interpretations. Designing demonstrations for new actions also often requires several trials; users often do not learn to anticipate an action until they have seen it occur several times.

System demonstration in Range

System demonstration is employed through the design of Range's features. In Range's transition from the ambient display to the drawing surface, for instance, the animated transition of the images and backdrop is a demonstration-of-action that calls more attention to the mode change than a sudden switch between modes would. This demonstrated transition also provides a handle for override, which will be discussed in the next section, allowing for more seamless negotiation of what board objects should stay active. Similarly, the movement of board objects in the making space feature needs to occur slowly and smoothly enough that users who detect a problem (ink strokes that are mis-clustered and hence do not move in concert, for instance) can more easily remedy the error.

Demonstration need not take place concurrently with the actions that are being demonstrated. In fact, the determination of good points to interrupt and alert the user to background actions are key to fluid interaction design [9]. When Range clusters ink strokes, the outlines for the clusters appear when the user steps back into move/selection mode. This act serves as system demonstration, as the outlines indicate that the mode has changed from sketching to editing, how the ink strokes have been clustered, and what strokes will shift in concert if moved. The timing of the interaction, however, prevents the system demonstration from distracting users when they are actively working on drawing at the whiteboard.

Override

Override techniques allow users to repair misinterpretation of the user's state, or to interrupt or stop the system from engaging in proactive action. This usually occurs after one of the previous two techniques (user reflection and system demonstration) alert the user to some inference or action that is undesirable. Although the two are often conflated, override is distinct from "undo" because it is targeted at countering the action of the system rather than reverting a command by the user.

The trajectories of overrides for user reflection and system presentation are illustrated in Figures 6 and 7. Overrides always start in the upper right hand quadrant (because users cannot repair perceptions or actions that they are unaware of) and move to the upper left hand quadrant, where the users are exerting explicit control.

Overrides are often easy to design intuitively, because users expect to be able to override things. At the point that users see some unwanted action taking place, they try numerous ways of trying to override the action; it is merely a matter of designing a ubicomp system so that the user's frantic override behaviors are registered as an input. It is possible for the designer to design in affordances for overrides—handles and edges, for example, that the user can grasp, or shields that the user can use to perform blocks. The wide array of potential affordances for override in ubiquitous computing environments can be a blessing or a curse for

physical interaction designers; it is important to test to make sure both that overrides are intuitive and that the number of potentially override-able actions presented at any time is limited so that the user is not overwhelmed.

Override in Range

The Range whiteboard demonstrates override capabilities in several places. In the transition between display and whiteboard, users can stop the transition by moving out of the proximal zone of active board use. Users are also able to "grab" digital content to use it as part of the whiteboard contents. They can also employ this grabbing technique to stop the motion of objects that are being moved to make space in the center of the board.

The design of feedback displays can double as handles for override; for instance, users who perceive a mis-clustering of ink strokes by the Range whiteboard can override Range's inferred clusters by redrawing the outline. The interaction cost of manipulation or correction is no more than it would be without the auto-clustering feature.

IMPLICATIONS FOR DESIGN AND DESIGN RESEARCH

Implicit interactions enable people to communicate efficiently, but understanding how to design effective implicit interactions requires more of designers than intuition about how to make things subtle or invisible. Indeed, in designing implicit interactions for the Range whiteboard we found that implicit interactions often rely on the counterintuitive strategy of calling attention to observed or inferred perceptions and prospective or on-going actions.

Our analysis of Range shows how the implicit interaction framework provides a significant contribution to prior models for implicit interaction [2]; without the key variable of initiative, it would not be possible to distinguish user reflection techniques from that of system demonstration techniques, or indeed, to map the role of override. In addition, this model helps to show important difference between designing static objects that utilize what a user already tacitly knows [24] and designing interactive objects that proactively engage the user; it is important for implicitly interactive objects to draw attention in a non-exclusive and time-sensitive fashion–something that designers of intuitive hammers do not have to worry about. Finally, this framework is generative, providing designers with patterns and templates that are crucial in creating low-cost response routines for "smart" systems that perform recognition and proactive action [13].

The implicit interaction framework helps designers to track and analyze interaction sequences along the critical interaction variables of attentional demand and initiative. The key implication of this framework for designers is that design solutions can be based on patterns of interactions rather than conventions of domain or context. This means that implicit interaction techniques developed for one domain can be generalized and applied analogously to another domain.

This, in turn, implies that design researchers might go beyond the profiling of context-specific aspects of various domains by studying interactions with an eye towards developing effective and generalizable interaction techniques. The strategies of user reflection, system demonstration and override are likely to have many more variations than those noted in this paper. Design researchers may be able to broader the understanding of implicit interactions further, discovering new interaction trajectories and characterizing their usage.

CONCLUSION

Implicit interactions are evident in the design of everything from automatic spell-checkers to interactive robots. By explicitly articulating how, when and why an interaction designer might use implicit interactions, we widen the designer's range in designing for challenging new domains such as ubiquitous computing.

In this paper, we have provided a proof-of-concept toward this goal by applying the implicit interaction framework to the design of an electronic whiteboard application, Range. Range's design is targeted specifically to the needs and practices of informal meeting participants, and yet the framework allows the interaction design techniques used in Range to be generalized to inform the design of implicit interactions in analogous domains.

This work provides a common basis for interaction designers to explore and share the range of implicit interactions and techniques. We provided a framework for better understanding the range of implicit interactions, and illustrated how implicit interaction techniques can be used to prevent, mitigate and correct the problems of proactivity in the area of electronic whiteboard design. The intent of this work is to provide interaction designers working a wide variety of disparate domain- and task-specific ubiquitous computing systems with a framework that allows them to build on each others' patterns and techniques. This can enable designers to better develop more sophisticated ways of implicitly interacting with systems in everyday life.

ACKNOWLEDGEMENTS

This research was supported by equipment grants from Intel Corporation and SMART Technologies, a fellowship from the Intel Foundation, as well as a grant from the Wallenberg Global Learning Network. Thanks to Terry Winograd, Larry Leifer, Paz Hilfiger-Pardo, Isabelle Kim, David Akers and Leila Takayama for their assistance.

REFERENCES

1. Bellotti, V., Back, M., Edwards, K.W., Grinter, R. Henderson, R., Lopes, C. Making Sense of Sensing Systems: Five Questions for Designers and Researchers. In *Proc. CHI 2002*, ACM Press (2002), 415-422.
2. Buxton, W. Integrating the Periphery and Context: A New Model of Telematics. In *Proc. Graphics Interface 1995*, (1995), 239-246.

3. Cavanagh, P. Attention routines and the Architecture of Selection. In *Cognitive Neuroscience of Attention*, Posner, Michael I, ed. Guilford Press (2004), 23-24.
4. Chase, W. G. and Simon, H. A. Perception in Chess. In *Cognitive Psychology* 4(1973), 55-81.
5. Chung, E., Hong, J.I., Lin, J. Prabaker, M.K., Landay, J.A., and Liu, A.L. Development and Evaluation of Emerging Design Patterns for Ubiquitous Computing. In *Proc. DIS 2004*, ACM Press (August 2004), 233-242.
6. Clark, H.H. and Brennan, S.E. Grounding in Communication. In *Perspectives on Socially Shared Cognition*, Resnick, L.B., Levine, J.M. and Teasley, S.D. eds., American Psychological Association (1991), 127-149.
7. Clark, Herbert H. Pointing and Placing. In *Pointing: Where Language, Culture and Cognition Meet.* Kita Sotaro, ed. Lawrence Erlbaum (2003), 243-268.
8. Elrod, S., Bruce, R., Gold, R., Goldberg, D., Halasz,F., Janssen,W., Lee, D., McCall, K., Pedersen,D., Pier, K., Tang, J., and Welch, B. Liveboard: A large interactive display supporting group meetings, presentations and remote collaboration. In *Proc. CHI 1992*, ACM Press (1992), 599-607.
9. Fogarty, J., Hudson, S.E. and Lai, J. Examining the robustness of sensor-based statistical models of human interruptibility. In *Proc. CHI 2004*, ACM Press, 207-214.
10. Guimbretiere, F., Stone, M. and Winograd, T. Fluid Interaction with High-Resolution Wall-Size Displays, In *Proc UIST 2001*, ACM Press (2001), 21-30.
11. Hall, E. *The Hidden Dimension.* Garden City, Doubleday, 1966.
12. Hartmann, B., Klemmer, S.R., Bernstein, M., and Mehta, N. d.tools: Visually Prototyping Physical UIs through Statecharts. In *Extended Abstracts of UIST 2005*, ACM Press (2005).
13. Horvitz, E., Kadie, C., Paek, T., and Hovel, D. 2003. Models of attention in computing and communication: from principles to applications. *In Communications of ACM* 46(3) ACM Press (March 2003), 52-59.
14. Ju, W., Ionescu, A., Neeley, L., and Winograd, T. Where the Wild Things Work: Capturing Physical Design Workspaces. In *Proc. of Conference on Computer Supported Cooperative Work 2004*, ACM Press (2004), 533-541.
15. Ju, W. and Leifer, L. The Design of Implicit Interactions: Making Interactive Systems Less Obnoxious. In *Design Issues*, 24(3) Summer 2008, 72-84.
16. Klemmer, S.R., Newman, M.W., Farrell, R., Bilezikjian, M., and Landay, J.A. The Designers' Outpost: A Tangible Interface for Collaborative Web Site Design. In *Proc UIST 2001*, ACM Press (2001), 1-10.
17. Landay, J.A and Myers, B.A. Interactive Sketching for the Early Stages of User Interface Design, In Proc. *CHI 1995*, ACM Press (1995), 43-50.
18. LaVie, N. Perceptual Load as a Necessary Condition for Selective Attention. *Journal of Experimental Psychology: Human Perception and Performance,* 21:3 (1995), 451-468.
19. Lin, J., Newman, M.W., Hong, J.I. and Landay, J.A, "DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design." In *Extended Abstracts of CHI 2000*, ACM Press (2000), 510-517.
20. McDonald, D.W., McCarthy, J.F., Soroczak, S., Nguyen, D.H. and Rashid, A.M. Proactive Displays: Supporting Awarenss in Fluid Social Environments. In *ACM Transactions on Computer-Human Interaction.* (14) 4, 16.
21. Moran, T. P., Chiu, P, van Melle, W. & Kurtenbach, G., Implicit Structures for Pen-Based Systems within a Freeform Interaction Paradigm. In *Proc. CHI 1995*, ACM Press (1995), 487-494.
22. Moran, T. P.; Chiu, P.; Van Melle, B. Finding and Using Implicit Structure in Human-organized Spatial Layouts of Information. In *Proc. CHI 1996*. ACM Press(1996) 346-353.
23. Mynatt, E. D., Igarashi, T., Edwards, W. K., and LaMarca, A. Flatland: New Dimensions in Office whiteboards. In *Proc CHI 1999*, ACM Press (1999), 346-353.
24. Pedersen, E.R. Tacit Interaction Talk in the Stanford University, Human-Computer Interaction Seminar series, May 14, 1999. Abstract online at: *http://kraka.com/DesignPortfolio/tacit.html*
25. Pedersen, E.R., McCall, K., Moran, T. and Halasz, F. Tivoli: An Electronic Whiteboard for Informal Workgroup Meetings. In *Proc. CHI 1993*, ACM Press (1993), 391-398.
26. Prante, T., Röcker, C., Streitz, N. A., Stenzel, R. , Magerkurth, C. , van Alphen, D. and Plewe, D. A. Hello.Wall –Beyond Ambient Displays. In *Adjunct Proceedings of Ubicomp 2003*. 277-278.
27. Russell, D.M., Trimble, J P., and Dieberger, A. The Use Patterns of Large, Interactive Display Surfaces: Case Studies of Media Design and Use for BlueBoard and MERBoard. In *Proc HICSS 2004*. IEEE (2004).
28. Streitz, N. A., Geißler, J., Holmer, T., Konomi, S. Müller-Tomfelde, C., Reischl, W. Rexroth, P., Seitz, P., and Steinmetz, R. i-LAND: An Interactive Landscape for Creativity and Innovation. In *Proc. CHI 1999*, ACM Press (1999), 120-121.
29. Tennenhouse, D. Proactive Computing. In *Communications of the ACM* 43:5, ACM Press (May 2000), 43-50.
30. Tolmie, P., Pycock, J., Diggins, T., Maclean, A. and Karsenty, A. Unremarkable Computing. In *Proc. CHI 2002.* ACM Press (April 2002), 399-406.
31. Vogel, D., and Balakrishnan, R. Interactive Public Ambient Displays: Transitioning from Implicit to Explicit, Public to Personal, Interaction with Multiple Users, In *Proc. UIST 2004*, ACM Press (2004), 137-146.
32. Weiser, M. The Computer for the 21st Century. *Scientific American*, 265:3 (September 1991), 94-104.
33. Weiser, M. Ubiquitous computing. In *IEEE Computer* 26, IEEE(October 1993), 71-72.
34. Weiser, M. and Brown, J.S. Center and periphery: Balancing the Bias of Digital Technology. In *Blueprint for the Digital Economy*, Tapscott, D. ed. McGraw-Hill (1998), 317-335.
35. Wroblewki, L. Visible Narratives: Understanding Visual Organization. On *Boxes and Arrows*, AIGA, January 20th, 2003.