Evaluating Intelligibility Usage and Usefulness in a Context-Aware Application

Brian Y. Lim, Anind K. Dey

March 8, 2011
CMU-HCII-12-103

Human-Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
{byl, anind}@cs.cmu.edu

This work was supported by the National Science Foundation under grant 0746428 and the author's National Science Scholarship (PhD) from the Agency for Science Technology And Research, Singapore. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author and do not necessarily reflect those of the funding agencies.

Keywords: intelligibility, explanation, user study, context-awareness, human-computer interaction.

ABSTRACT

Intelligibility has been proposed to help end-users understand context-aware applications with their complex inference and implicit sensing. Usable explanations can be generated and designed to improve user understanding. However, will users be willing to use these intelligibility features? How much intelligibility will they use, and will this be sufficient to effectively improve their understanding? We present a quasi-field experiment of how participants used the intelligibility features of a fully-functional intelligible context-aware application. We investigated how many explanations they willingly viewed, and how that affected their understanding of the application's behavior, and suggestions they had for improving its behavior. We discuss what constitutes successful intelligibility usage, & provide recommendations for designing intelligibility to promote its effective use.

INTRODUCTION

Context-aware applications use implicit sensing and complex inference to automatically and calmly adapt to serve users [3, 20]. Lay end-users may not be aware of what these applications know, and struggle to understand their behaviors. This can lead to user frustration and loss of trust in the applications [13]. To counter this, context-aware applications should be *intelligible* (also called transparent, comprehensible, scrutable) by providing explanations of their behavior [1]. Indeed, there have already been several context-aware applications that support some level of intelligibility (*e.g.*, [2, 17, 18, 19, 22]). These systems support a limited set of explanations users can ask for: What, Certainty, Inputs, Why, and Why Not. However, Lim & Dey [7] found that users ask a wider range of questions of context-aware applications, and that different explanations have different impacts on user understanding [6]. To support this wider range of explanations, Lim & Dey [9] designed Laʞsa, which provides explanations to 8 question types for several context types. While that work provides a crucial step for designing intelligibility to be more usable and interpretable, it stopped short of evaluating the impact of intelligibility on users. Lim & Dey [10] investigated the impact of intelligibility on understanding and impression, but this was studied with questionnaires and 'paper' prototypes rather than an interactive prototype. Furthermore, intelligibility was shown "always on" to participants, so they were biased to look at the explanations. This leaves open the research questions: even if intelligibility can improve user understanding and trust, will users want to use it, and, if so, how much? Moreover, given how much they do use, how much will that improve their understanding of context-aware applications?

Related work has explored the impact of explanations on end-users as they used context-aware systems. Rukzio *et al.* [17] evaluated a mobile phone automatic form filler in a lab study, and found that *"visualizing the uncertainty of the system was mostly not used nor was it helpful."* Tullio *et al.* [18] evaluated an intelligible interruption door display over six weeks, and found that users were able to *"attribute concepts of machine learning to their system,"* but had difficulty remembering relevant features. Cheverst *et al.* [2] deployed the Intelligent Office System that provided explanation visualizations of rules and confidence. However, regarding explanations, their evaluation focused on eliciting user preference about visualization format, and not on their impact. Welbourne *et al.* [22] investigated the use of Panoramic that is able to explain location with timeline visualizations. However, their evaluation involved participants investigating realistic, but fictitious, data. Vermeulen *et al.* [19] conducted a pilot user study of PervasiveCrystal in a simulated museum with five participants, who *"were able to use the questions interface to find the cause of events"* of three tasks.

We add to this body of work evaluating intelligible context-aware systems by explicitly measuring the *usage* of intelligibility in a high-fidelity prototype that provides over 9 explanation types (*e.g.*, Certainty, Why, Why Not, What If) for three context types (Availability, Place, Sound). We iterate on Laʞsa [9] to investigate usage under realistic situations with real-time application behavior and generated explanations. We also investigate the impact of this usage on user understanding of the application's inference. Our contributions show intelligibility is useful by investigating:

1. How much participants *use* intelligibility in a real context-aware application,
2. Their *opinion* of the *usefulness* of the explanations to understand application behavior and situations, and
3. How *useful* their use of intelligibility is on understanding and handling of these situations.

The rest of the paper is organized as follows: we articulate our objective to explore the usage of intelligibility, and our hypothesis that increased intelligibility usage will improve user understanding. We developed a functional intelligible context-aware prototype for this study, which we describe next. Following that, we elaborate on the quasi-field experiment we conducted, where participants engaged in "everyday" scenarios *in-situ* using the prototype. We follow this with the results showing how participants used intelligibility and how that improved their understanding of application inference. Finally, we discuss design implications due usage patterns and constraints, and how to encourage users to use more intelligibility to further improve their understanding.

OBJECTIVES AND APPROACH

We have two objectives for this study: one explorative, and another hypothesis-driven.

1) Exploring the usage of Intelligibility. We aimed to investigate *how* users use intelligibility when facing different scenarios, *how much* they use, and for *how long*.
2) Hypothesis: Increased usage of Intelligibility will improve user understanding. We hypothesize that using intelligibility more will help users better understand application inferences and the current situation.

To accomplish these objectives, we conducted a quasi-field study where participants used a fully interactive, intelligible context-aware application under real-world, "everyday" situations (similar to [16]). To improve ecological validity of our results, we minimized interference from the experimenter by using logging and analyzing UI events [4] of intelligibility usage (without thinking aloud), and post-incident interviews. This experimental set-up strikes a balance between controlling for critical incidences, and allowing participants to use intelligibility naturalistically. Note that in this work, we do not claim to cover a comprehensive set of situations or motivations under which intelligibility may or may not be used significantly. However, we seek to gain an initial insight into how intelligibility may be used in a context-aware application reacting to a real physical environment.

We next describe the intelligible context-aware prototype we developed and employed to study intelligibility usage.

LAKSA 2 PROTOTYPE

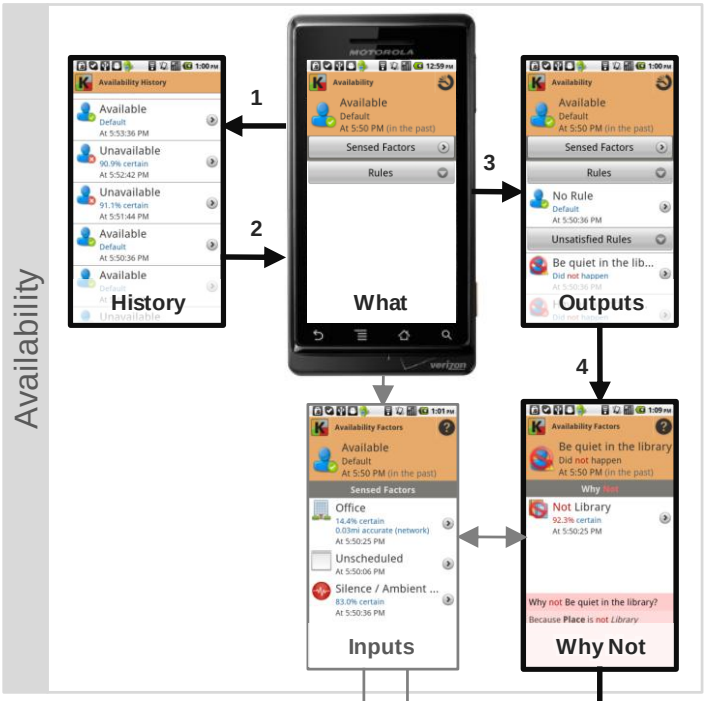
Mobile phones allow people to keep in touch with others and be easily reachable. However, there are times when receiving calls are inappropriate, as they are socially disruptive (*e.g.*, in meetings and movie theatres), or they interrupt productive work. Users can manually silence their phones, but they may forget to reset their phones to ring again afterwards [12]. Hence, it will be useful if the phone can automatically set the ringer mode (*e.g.*, [5, 15]). Also targeting this compelling application problem domain, we have developed Laʁsa 2, a mobile application that senses various contexts (Place, Sound, and Schedule) about the user to automatically infer her Availability, and set her phone's ringer mode. Our focus is the use of Laʁsa as a platform to explore the use of intelligibility in a context-aware application. Next we describe Laʁsa's contexts.

Availability: *Available, Semi-Available, Unavailable* — is inferred from rules regarding the following three factors.

Place: *Office, Café, Library, etc.* — represents the semantic location of the user. It is inferred by sensing latitude and longitude from the Android Location API (uses GPS, Wi-Fi, and cell tower positioning), and matching to pre-specified named places. The user's sensed location is modeled as a radial Gaussian, with decreasing likelihood further away from the latitude and longitude coordinates (as in [9]). Laʁsa stores a list of named places with coordinates and size (circle radius) to compare against to infer whether the user could be at each place. Each Place is inferred with different certainty based on how much the user's estimated location area "overlaps" with the area of the named place: more overlap leads to higher certainty.

Sound: *Talking, Music, and Ambient Noise* — represents the sound activity that Laʁsa recognizes from what the phone's microphone hears. Inferences come from a naïve Bayes classifier trained on sound samples. Features extracted are similar to [11]: *e.g.*, mean of power, low-energy frame rate, spectral flux, and bandwidth. These are renamed to lay terms that end-users can understand.

Schedule: *Personal, Work, Unscheduled* or *Other Event*.



Walkthrough of troubleshooting S3

After getting an audible call in the Library, a participant may follow these steps to troubleshoot why Laʁsa did not silence the phone.

0. After turning the screen on, she will see the current Availability inference as a **What** explanation.
1. She can investigate about a specific past event with **History**.
2. After selecting the desired event, in this case, one at 5:50:36 PM, she sees the **What** explanation of her Availability at that time.
3. She can see which rules were triggered and which were unsatisfied via the **Outputs** explanation.
4. On selecting an unsatisfied rule, she sees a **Why Not** explanation indicating that Laʁsa did not think she was at the Library.
5. She can dig deeper and ask about the Place inference to see the **What** explanation indicating Laʁsa thought she was at the Office with 14.4% **Certainty**.
6. On inspecting the possible places with the **Outputs** explanation, she can see Library was its second choice.
7. She can inspect **Why** she was inferred to be at the Office, and see that her sensed location (concentric blue bubbles) overlaps a lot with the Office bubble (green);
8. Go back, and
9. Inspect **Why** the Library had a lower **Certainty** (because the Library bubble was too small, while the Office bubble was too big).

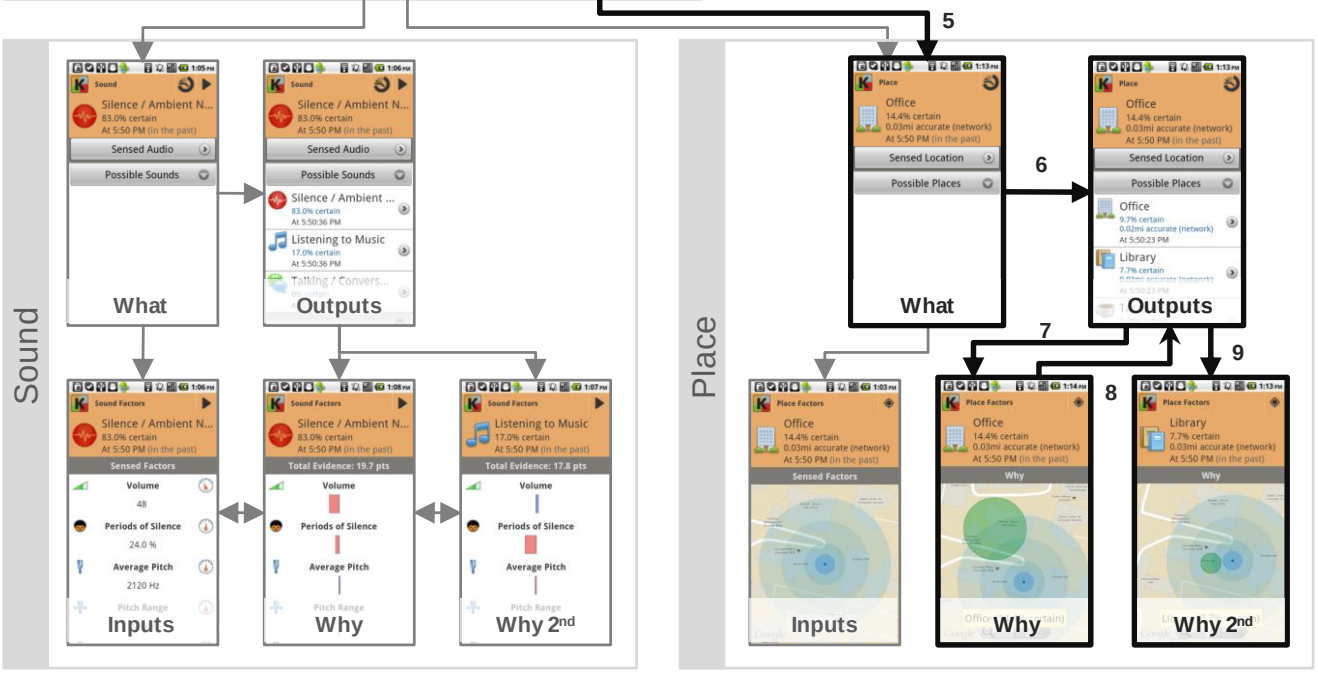


Figure 1. Screenshots of the Laʁsa application showing several explanation types of the upper-tier context Availability, and lower-tier contexts Sound and Place. Arrows between each screenshot shows how a user can transition from one explanation to another. The bold trace indicates how a participant may explore the intelligibility features in 9 steps to troubleshoot Scenario 3 about the phone ringing in the Library. The Intelligibility UI was adapted from [9, 10] using the "bubbles" metaphor to explain how Place is inferred, and the "weights of evidence" bar charts to explain how features vote for different Sounds.

Intelligibility Features

Having defined the context types, we make Lakṣa intelligible so that users can understand what it knows and how it makes inferences. Lakṣa provides explanations recommended by [7]:

1. **What** is the inference for the context? With how much **Certainty**? **When** was this value inferred?
2. **History**: what was the inference at time *H*?
3. **Inputs**: what details affect this context? (Factors, input features, related details, *etc*.)
4. **Outputs**: what values can this context be inferred as? With how much **Certainties** are these values inferred?
5. **Why** was this value inferred?
6. **Why Not (Alt)**: why wasn't this inferred as *Y*, instead?
7. **What if** the factors are different, what would this inference be? (Requires user manipulation)
8. **Description**: meaning of the context terms and values.
9. **Situation** of what was happening to affect the inference to provide a ground truth of what was being inferred, *e.g.*, playing an audio clip of what was heard.

Some explanation types have been aggregated to reduce the number of questions users need to ask (*e.g.*, What + Certainty + When, Outputs + Certainties). For simplicity, What If was only provided for Availability. Also, Schedule does not have particularly expressive explanations, because it is easy to understand calendars and events.

Design Iterations and Updates

While this iteration bears many similarities to the original Lakṣa prototype [9], its design and functionality has been significantly refined and it uses streamlined questioning to be simpler for users. Further feedback from colleagues, who are HCI researchers, helped make the user interaction more consistent throughout the application, and reduced the application functionality so that users can grasp its concepts within a 2-hour study. Therefore, we removed the Motion context of [9]. Lakṣa 2 also supports more explanation types that are relevant to realistic use, such as History and Situation. Finally, Lakṣa 2 is not a social-awareness application like its predecessor, and it was deployed on a mobile phone instead of a Tablet PC.

Implementation and User Interface

We developed Lakṣa for Android 2.2 Froyo, and deployed it on the Motorola Droid for the user study. Sensing for location, calendar events, and microphone audio were performed using background services on the phone every 30 seconds. Higher-level inferences for Place, Schedule, Sound, and Availability are computed in the background, in response to each sensed instance. To recognize sounds, we used a port of Weka for Android [21]. We also partially ported the Intelligibility Toolkit [8] to Android, to support the *querying* for various questions, *generation* and *reduction* of explanations about the contexts, and *presentation* of the explanations in various graphical and textual formats. Unlike [9], the Lakṣa 2 prototype is fully implemented on the mobile phone for sensing, inferring, reacting, and displaying explanations. Figure 1 shows several screenshots of the Lakṣa 2 prototype with a walkthrough example of how to use it. Each explanation is viewed as a *page view*. Users can transition from one to another by clicking on buttons, menu items (from the options menu), and flinging (swiping). Some explanations allow scrolling to see more details.

SCENARIO-DRIVEN QUASI-FIELD STUDY

To explore the use of the intelligibility features in Lakṣa, we conducted a controlled scenario-driven user study, where participants encountered situations that may arise with the use of Lakṣa. We were interested in whether and how participants used the intelligibility features to understand the application behavior. We conducted a quasi-field study rather than a field deployment to (i) present participants with controlled critical incidences, and (ii) observe and measure their subsequent behaviors due to these incidences. It otherwise would have been difficult to know when critical incidences occurred in the field, or why.

Procedure

An experimenter first briefed the participant about the study, and presented her with printed instructions. These describe how to use the Android phone, Skype (for receiving or checking calls), and Lᴀksa's functionality and interface. The experimenter gave a walkthrough of Lᴀksa, demonstrating its features and how to interpret them. We provided participants with availability rules that Lᴀksa was pre-programmed with: four rules setting availability to Unavailable and Semi, and any other case as Available (*e.g.*, if the user is in her office and Lᴀksa hears talking, her status is set to Unavailable). Participants did not touch the phone until the first scenario (S1).

The participant was instructed that she works with equally ranked coworkers in several offices, and that they work together on a team project. She was provided with the following motivation: she needs to evaluate Lᴀksa as a newly acquired application, which can improve her team's productivity by moderating interruptions. She is tasked with the overall goal of determining when Lᴀksa behaved appropriately or not, and figuring out how to improve its future behavior by (i) editing availability rules, (ii) changing lower-level settings (*e.g.*, size of Place bubbles), or (iii) changing behavior (*e.g.*, lower the music volume). She would also be responsible for subsequently teaching her coworkers how to best configure Lᴀksa. After reading the instructions, the participant begins the scenarios.

Controlled In-Situ Scenarios

The user study was scenario-driven to expose participants to situations they may encounter with Lᴀksa. To increase the visceral quality of the scenarios, each scenario is set up by bringing the participant to the necessary places (Office, Library, or Café) and asking her to carry out an initial task, *e.g.*, looking for a library book (S3). The experimenter shadowed the participant for every scenario. The participant engaged in the activity for a few minutes to become absorbed in the situation. Critical incidences were triggered by the experimenter calling the participant's phone (if necessary), and presenting her with a printed flash card describing what was happening, and any associated dialog with coworkers during the phone call. The participant was free to interact with Lᴀksa as much or as little as she wished, and prompted to *not* think aloud. For each scenario, after she was done looking at Lᴀksa, she turned off the screen, and the experimenter conducted a structured interview with audio recording. She was asked about her opinion of the situation and the application, her understanding of how Lᴀksa was making inferences, and any suggestion she may have for improving its behavior.

We employed four scenarios to span three situational dimensions: (i) Exploration / Verification (S1) of Lᴀksa's functionality and explanations; (ii) Fault Finding (S2, S3), where Lᴀksa behaved inappropriately, and participants had to troubleshoot it; and (iii) Preemptive Exploration (S4) where participants investigated a potential future situation.

S1: Talking in the office. Training session where the participant learned Lᴀksa's core features and explanations. She could explore Lᴀksa as much as possible to familiarize herself with the application UI and explanations.

S2: Missed call while reading news & listening to music. The participant is asked to read any news articles they fancy from www.cnn.com while they listen to a song (Sound of Silence[1]) through the computer speakers. Meanwhile, she received multiple calls from a coworker, but she misses the first few calls. The experimenter actually called the participant's phone but it did not ring the first few tries since Lᴀksa automatically silenced its ringer. Near or at the end of the song, the phone would ring again and the participant would notice the call. Through a flash card, the participant learned that her coworker, Damien, was frustrated from trying to call her repeatedly over the past three minutes; he would like her to check her email, and fix her phone. The email pertained to finding a library book to review for their shared project.

[1] Sound of Silence by Simon and Garfunkel. http://youtu.be/eZGWQauQOAQ. Retrieved 8 March 2012.

Lᴀksa had mis-inferred Sound as Talking instead of Music, and behaved inappropriately by silencing the phone.

S3: Phone interruption in the library. As a follow-up to Damien's email, the participant walked to a nearby library to search for the book, and read it. Meanwhile, the experimenter called her phone again, causing it to ring audibly in the quiet library. This simulated a coworker calling. Lᴀksa had mis-inferred the participant's Place as still in the Office instead of Library, and behaved inappropriately by allowing the phone to ring.

S4: Preemptively checking availability in café. The participant received a flash card describing that she frequents a café (in a nearby building), and should check whether she will be able to receive calls there. Participants were not prompted what to do to achieve this objective.

MEASURES & DATA PREPARATION

We are interested in measuring how useful intelligibility was for the participants in terms of how much they used, and how that impacted their understanding of Lᴀksa, and their suggestions on how to control it to resolve any issues.

Usage of Intelligibility

To measure intelligibility usage, we logged when participants viewed each explanation page in the UI. This allowed us to measure, for each scenario, which explanation types each participant viewed, how many (**# Explanation Types**), when they were viewed, for how long (**Duration**), how often (**View Count**), and their sequence order (**Step** number). We built a network graph for each participant scenario to illustrate the *sequence diagram* of how he used intelligibility (*e.g.*, Figure 1). With these we can observe general patterns of use, and identify errors in the logging. Since participants may view certain pages only to get to another page (*e.g.*, transitioning through Availability Inputs to get to explanations about Place), they may only view them for a very brief duration. Hence, we filtered out views with durations < 1 second. Additionally, intelligibility usage patterns may also affect what a user learns of the application. One such metric is the **Context Ratio** of how many explanation types of deeper contexts (Place and Sound) are viewed compared to that of the shallower context (Availability). Having generated these metrics, we wanted to investigate if they influence user understanding.

User Understanding and Suggestions for Control

We measure how well participants understood the application behavior and scenario circumstance by transcribing audio from interviews along with notes. To measure their understanding, we asked them what they understood about what Lᴀksa knew and how it was reasoning. The transcript was coded into units of *beliefs* to characterize their mental models, using the coding scheme in Table 1. Their statements are a lower bound of their understanding, since they may not have said everything they believed. To derive a single metric of understanding, an **Understanding Score** is calculated for each participant scenario by adding all 7 codes for both Place and Sound (Max=14). This score represents the breadth and depth of understanding a user has for the scenario.

Another measure of how well participants understood Lᴀksa is how many effective **Control Suggestions** they provided to overcome any issues or problems in the scenarios. A weighted **Control Score** is calculated from summing scores for each code in the scheme in Table 2. This score represents the number and effectiveness of suggestions provided for the scenario. Partially effective suggestions are given only half a score. These suggestions may have side-effects that compromise application performance in other situations (*e.g.*, adjusting the weights of a sound factor to influence recognition).

Code		κ		Description / Example Transcripts
		Place	**Sound**	
U_1	Value	.83	.89	Indicated knowledge of the inferred value of the factor.
U_2	Alternative Values	.78	.85	Indicated knowledge of other inferred (2nd, 3rd, *etc*.) or uninferred values. Compared different values that were inferred differently, e.g., P01S2 *"Talking (evidence=85.4) very close to music (84.?). Could have gone any way."*
U_3	Certainty	.94	.87	Described certainty of inferred value, *e.g.*, P02S3: *"It was 9.3% certain I was at the Office"*; P03S3: *"blue bubbles were too big."*
U_4	Inputs	.85	.94	Mentioned at least one input feature / factor of the context, e.g., Pitch, Periods of Silence, *"the blue bubble was directly over the Library building."*
U_5	Model	.86	1	Described the mechanism for inferring the factor, *e.g.*, P17S3: *"It looked like it was actually probably closer to the library, but since the library bubble was very small then it calculated the probability was very low."*
U_6	Technical	.90	0*	Provided a deep technical mechanism for the inference not explicitly described in the explanations, *e.g.*, P18S3: *"It seems to be based on its Wi-Fi connection, and … because it said networking and it gave the location badly and we're deep inside a bunch of concrete and metal, so the GPS shouldn't be working right now."*
U_7	Situation Justification	.80	.94	Provided a situational justification for the phone's inference that was not from the intelligibility UI, *e.g.*, P02S2: *"The music was much more mellow, and they were really singing"*; P07S3: *"We were very close to previous location [Office], not easy to pinpoint current place [Library]."*

Table 1: Coding scheme for user understanding. Participants' mental models were decomposed into beliefs based on what they *explicitly* said and *tacitly* implied. Each scenario may have multiple codes, each either 0 or 1 indicating whether the correct corresponding beliefs were expressed. We only coded for Place and Sound factors, since participants' understanding of Availability can be derived from their understanding of these. Inter-coder reliabilities (κ) for each code were calculated with a 35% random sample of the scenarios by a second coder. ** denotes apparent low reliability due to low count.*

Code		κ	Description / Example Transcripts
C_1	Availability Rules	.89	Proposed a new rule, editing an existing rule, or deleting one, *e.g.*, delete rule "Someone's Talking"; add rule "Office + Music → Vibrate"
C_2	Place Settings	.90	Suggested to adjust the bubbles of Places by enlarging, shrinking, or moving them. Suggested to threshold blue bubbles to calculate overlap between sensed location and Places.
C_3	Sound Settings	.89	Suggested to adjust a feature weight to tweak inference; Suggested to expand training data, *e.g.*, P10S2: "Teach it more about music by inserting iTunes catalog."
C_4	Change Behavior	.92	Proposed to change behavior to ameliorate problems in the scenario, *e.g.*, "Reduce the volume of music"; "Have phone screen facing up (when on table) so that it will be visible when it lights up during a call"

Table 2: Coding scheme for control suggestions. Participants' suggestions to improve Lαksa for each scenario were coded with values: 0=Ineffective, 0.5=Partially effective, 1=Effective. Each code may be counted multiple times depending on how many suggestions were made. Inter-coder reliabilities (κ) were calculated with a 50% random sample by a second coder.

Perception of Application and Explanations

We were interested in how participants perceived Lᴀksa and its explanations for each scenario. As a manipulation check of the scenario designs, we asked participants how they perceived Lᴀksa's **Behavior Appropriateness** (7-point Likert scale: -3=Strongly Inappropriate, 3=Strongly Appropriate). We also asked if they agreed or disagreed that the explanations were helpful (**Explanation Helpfulness**; 7-point: -3=Strongly Disagree, 3=Strongly Agree).

Next, we describe how participants used intelligibility, and how that impacted their understanding. We treated S1 as a warm up for participants and excluded its results from our analyses.

RESULTS

Using a local recruiting website, we recruited 19 participants (11 females) with ages 19 to 65 (Median=26) years. We dropped P13 because he did not continue beyond S2, and did not understand the scenarios well. 9 participants were graduate students, and three were undergraduates. P01, P16, and P17 were students in a computer-related field (Electrical and Computer Engineering, Software Engineering, and Learning Technology). P18 was a web programmer, while the others spanned a wide range of areas (*e.g.*, actor, pianist, field interviewer, hospital administrator, chemical engineering, retiree). We engaged each participant for 1h 44min on average (range: 1h 29m to 1h 58m). Each participant was compensated $20.

While we strove to make all user experiences consistent for the experiment, we also strove to have Lᴀksa behave faithfully to the scenarios the participants were situated in, and what they did. Consequently, there was some variability in what Lᴀksa sensed and the resulting explanations. For example, location sensing accuracy depended on where the participant decided to walk to, weather conditions and other environmental factors affecting signal strength; when the participant walks to the café in S4, she may hear background music, or find a seat nearby people who are talking.

For S4, participants exhibited two distinct behaviors to explore the hypothetical situation: they either just sat where they were and tried to use the What If explanation facility (S4-if, 10 cases), or walked to the café to test Lᴀksa *in-situ* (S4-situ, 11 cases; some participants did both). Hence, we treat these as distinct scenarios.

In this section, we report results of participants' perception of Lᴀksa's behavior and explanations, how they used intelligibility, their understanding of Lᴀksa's behavior, and how their usage affected their understanding. We supplement the quantitative data with descriptions of what participants did and said, and provide interpretations.

Perception of Application Behavior and Explanations

As expected, participants perceived Lᴀksa's behavior was perceived as inappropriate for S2 and S3, but appropriate for S4 ($F_{3,25}$=3.90, $p<.05$; contrast test: $p<.01$). Participants generally found the explanations helpful ($F_{3,25}$=3.90, $p<.05$), but this depended on the Scenario. Explanations were less helpful in S2 than in S4 (Tukey HSD test: $p<.05$).

Next, we characterize how participants used intelligibility: how often they looked at explanations, and for how long.

Intelligibility Usage

From usage logs combined across S2 to S4, we determined participants' overall usage of intelligibility (see in Table 3), and their usage for each explanation type (see Table 4). Most participants actively looked at many Explanation Types (Median=8), many times (View Count Median=21), for about 3 minutes per scenario. This suggests they valued intelligibility enough to use it. They also tended to look more at deeper contexts (Place or Sound) than just Availability (Context Ratio Median=1.4). Some participants were very engaged in

using intelligibility (View Count Max=65, Scenario Duration Max=12.5min), while some were conservative: min 2 views (P08S4-if), 1 explanation type (P14S4-situ), scenario duration <1 min (P08S4-if), or not looking at deeper contexts (Context Ratio=0, P02S2, 7 participants for S4-if, P05S4-situ, P14S4-situ).

From Table 4 Left, we identify which explanation types were more popular, *i.e.*, higher view count. The Availability What explanation was the first page that participants saw when they turned the screen on, so it has the highest count. Availability Inputs is also high because most participants used it as a gateway to see explanations of deeper contexts. Availability History was popular because participants had to ask about specific events in the past. Participants viewed Outputs to see the expected inferences that were not made (particularly for Availability, and Sound). Although participants seldom viewed Definitions, they did so more for Sound because they were less familiar with its concepts. Due to the temporal nature of Sound, 9 participants played audio clips of what Laʁsa heard (Situation). 9 participants also used the Refresh function 30 times in total to get immediate feedback about the inference. They used it mostly during S3 and S4-situ, about Availability and Place, and for What, Inputs explanation types.

Per Scenario	Mean	SD	Std. Err.	Min	Median	Max
Steps (unfiltered)	27	16	2	2	23	71
View Count	24	15	2	2	21	65
# Explanation Types	7.9	3.4	0.4	1	8	19
Context Ratio	1.8	1.8	0.2	0	1.4	8
Total Duration (s)	205	136	18	52	196	749

Table 3. Summary statistics of intelligibility usage by participant scenario.

	Total View Count			**Median Duration (s)**		
	Availability	Place	Sound	Availability	Place	Sound
What + Certainty	232	114	69	5.7	3.1	3.2
History	130	5	3	7.5	-	-
Outputs + Certainty	84	102	33	6.4	5.6	4.9
Inputs	217	45	60	7.1	6.4	6.0
Why	26	16	44	3.5	9.9	6.1
Why Alt	21	35	41	4.7	9.6	4.9
What If	31	-	-	24.8	-	-
Definition	5	7	28	-	-	6.1
Situation	-	-	15			

Table 4. Usage of explanation types: total view count of explanation types for all participant scenarios, and median durations for respective views (for Total View Count > 15). Mean View Count per scenario can be calculated by dividing by number of participant scenarios, N=57.

We can also see how much time participants spent looking at each explanation type (Table 4, Right). While What If was not used often, when it was, participants spent significant time with it. This is because it is an interactive facility rather than a static display. For the other explanation types, participants on average spent less than 10 seconds viewing them, and may even view them as quickly as about 3 seconds. Furthermore, participants spent more time on explanation types that were more complex or contained more information (*e.g.*, Availability Inputs > Why, Place Why > Inputs, Sound Inputs > What).

Correlations between Intelligibility Usage and its Impact

Given the variation in intelligibility usage, we next explored how that affected participant understanding, control suggestions, and perception. We calculated correlations between our metrics of intelligibility usage, user understanding, control suggestions, and perception (see Table 5). These suggest some relationships, which we interpret. When participants perceived the application as behaving *less* appropriately, they viewed more explanations (a) and more types (b), spent more time exploring explanations (c), provided more suggestions for controlling and fixing the behavior (d), but perceived explanations as less helpful (e). They had higher Understanding scores when they viewed more explanations (f, h), viewed more about deeper contexts than shallower ones (i), or spent more time looking at explanations (j). The same was true for their Control score, perhaps due to their improved understanding (k). Strangely, explanation helpfulness was not correlated with intelligibility usage (g), and participants who perceived explanations as more helpful had fewer suggestions for effective control (l).

	Usage				Impact		
R^2	View Count	# Explanations	Context Ratio	Total Duration	Understanding	Control	Helpfulness
Appropriateness	-.37 [a]	-.32 [b]	-.02	-.41 [c]	-.11	-.34 [d]	.40 [e]
View Count		.81	.20	.63	.43 [f]	.29	-.17 [g]
# Explanation Types			.26	.45	.36 [h]	.30	-.15
Context Ratio				.06	.42 [i]	.30	-.05
Scenario Duration					.20 [j]	.13	-.08
Understanding						.41 [k]	.05
Control							-.23 [l]

Table 5. Correlations (Pearson's r) between usage of, and impact due to intelligibility. Significant correlations underlined. Some interpretations in text passage.

Now that we see some potential relationships between intelligibility usage and its impact, let us explore how well participants understood Laʞsa, and conduct statistical tests of whether and how usage affects understanding.

User Understanding and Control Suggestions

We first report the average understanding participants had. For each scenario, participants articulated 0 to 8 correct beliefs about Laʞsa's behavior (Median=4). Figure 2 (Left) shows the distribution of their correct beliefs. 41% of the beliefs were about the awareness of the inferred Value for Place and Sound, 28% about a broader understanding of the inference (Alternative Values and Certainty), 15% about the Inputs state and Model mechanism, and 2.5% about deeper Technical details. 14% of the beliefs were drawn from the situation to justify Laʞsa's behavior. While not coded, some participants expressed incorrect mental models, such as believing that the Place inference influences Sound inference, and vice versa (*e.g.*, P14-S2: *"[Laʞsa] infers location first [Office], then uses that to infer sound is likely talking [, instead of music]"*).

Participants provided 0 to 6 correct Control Suggestions (Median=2) for each scenario, and had an average Control Score of 2.10 (Std Err=0.29). This is significantly greater than 1 (*i.e.*, H_0: Score>1, p<.01). Figure 3 (Left) shows the distribution of effective and partial Control Suggestions to improve Laʞsa's behavior: Availability Rules (29%), Settings (27% Place, 8% Sound), Behavior Change (36%).

Regardless of how participants used intelligibility, they had non-zero Understanding and Control scores. However, the extent and pattern of intelligibility usage did affect how well participants understood Laʞsa, as we shall see next.

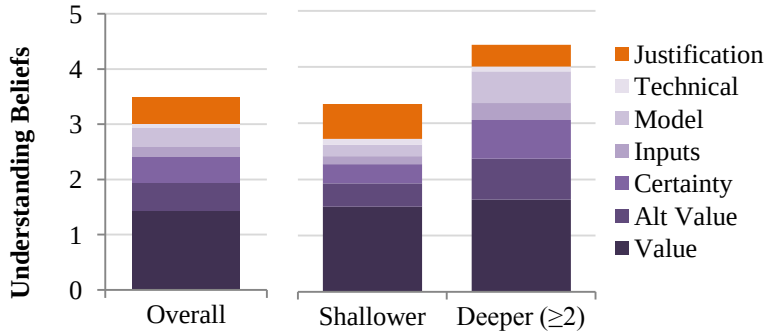


Figure 2. Distribution of belief types of Understanding overall, and by Context Ratio; normalized per scenario. Similar distribution for low View Count (<30) *vs.* high.

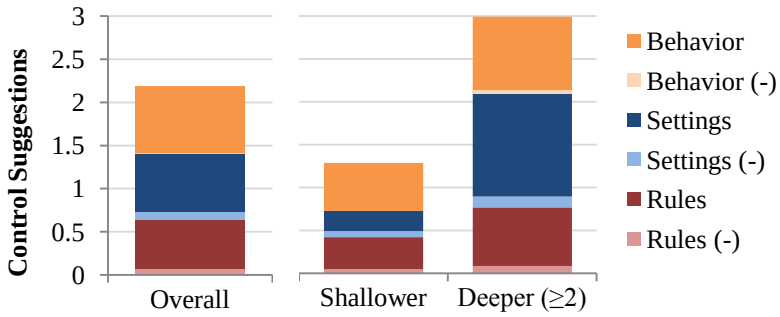


Figure 3. Distribution of effective suggestions participants made to improve Lᴀᴋsa's behavior; normalized for each scenario. (-) denotes partially effective suggestions with side-effects. Note this is not the weighted Control Score.

Impact of Intelligibility Usage on Understanding

From the correlations between usage and impact (Table 5), we chose View Count and Context Ratio as factors of intelligibility usage. We split View Count into discrete intervals of 10 counts (sample size: 9-16); we split Context Ratio into two groups Shallower (N=34) and Deeper (N=20), where participants saw twice as many explanations about Place or Sound than Availability (ratio ≥2).

We performed a mixed-model analysis of variance with Participant as the random effect, nested in Scenario, and View Count and Context Ratio as main effects. We fit separate models for Understanding Score (R^2=.434) and Control Score as dependent variables (R^2=.418). For Understanding Score, we found a marginal difference across View Count groups ($F_{4,47}$=2.20, p=.10; see Figure 4), and a contrast test found that when View Count <30 instead of ≥30, Understanding Score was lower (p<.05). Moreover, the score was higher when participants viewed explanations of Deeper contexts ≥2 times more than Shallower ones ($F_{1,47}$=4.20, p<.05; see Figure 5, Left). For Control Score, there was no difference across View Count groups, but it was higher for Deeper context ratios than Shallower ones ($F_{1,47}$=8.00, p<.01, see Figure 5, Right).

Intelligibility usage also affected the depth of understanding participants expressed. Figure 2 (Right) shows that when participants had deeper Context Ratios, they described 2.0 times more details about the factor inference (Alternative Values, Certainty, Inputs, Model, and Technical), but mentioned fewer Situation Justifications. Similarly, they provided 2.2 times more types of Control Suggestions, especially about Settings (Figure 3, Right).

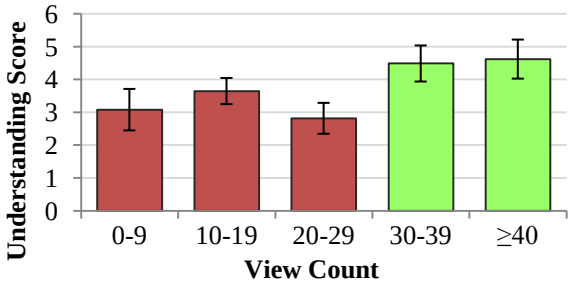


Figure 4. Participants had a higher Understanding Score when they viewed ≥30 explanations than more (p<.05, Left).

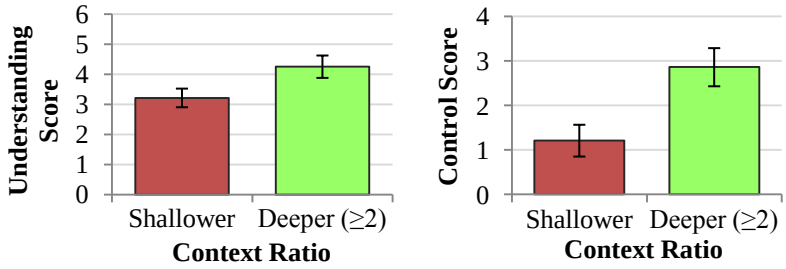


Figure 5. When participants ask more explanations about Deeper contexts (Place and Sound) than Availability, Understanding Score is higher (p<.05, Left) and Control Score is higher (p<.05, Right).

In summary, our results show how participants were willing to use intelligibility, and how quickly or deeply they used it. This satisfies our hypothesis that more Intelligibility Usage (View Count and Context Ratio) improves Understanding.

DISCUSSION, IMPLICATIONS, AND RECOMMENDATIONS

We discuss what we have learned about how intelligibility is used, and how that affects user understanding of context-aware applications. These have implications on how intelligibility should be provided, and how we should design intelligibility to facilitate its more effective use.

Usage and Usefulness of Intelligibility

By the extent that our participants viewed the explanations, we can conclude that intelligibility was *useful* for them to (i) engage with intelligibility (some participants deeply so), (ii) rate explanations as helpful, and (iii) gain better understanding of application behavior. We next discuss how they used intelligibility, and how certain usage patterns were more effective in improving user understanding.

Diverse Usage of Explanation Types

Participants used a diverse range of explanation types (see Table 4) and in diverse ways. What and Inputs were conduits to other explanations for participants to learn deeper reasons. However, although some explanation types were used less than others, participants viewed them for longer durations when they did (*e.g.*, Place Why / Alt). Furthermore, as with [9], the sequence diagrams of our participants revealed a variety of usage styles (*e.g.*, quick comparison between Why and Why Alt reasons, diving into a deeper context after going straight to Availability Inputs).

Unlike what was found in [6], our participants felt that the What If explanation was easy to use and liked it (*e.g.*, P11S1: *"[Using] it was just more fun … I like to think of hypothetical things, but it also gives me a sense of what the phone is capable of, and helps to develop trust when you know what to expect"*). In fact, for

S4, 10 participants chose to ask What If instead of immediately walking to the café. However, this fascination with What If can also give users false trust since it obscures potential pitfalls in sensing. Participants who used What If in S4 may not realize how noisy the café may be or that the Place inference was not particularly good there. P11 did not bother to explore Lʌksa's inference in-situ because *"technology is supposed to make your life easier; you shouldn't have to waste time to make sure it works right."* Perhaps providing warnings that sensing can fluctuate due to environmental conditions may help users be more careful when using What If.

While not explicitly an explanation, Refresh was used to understand what Lʌksa was sensing and inferring in the moment. For S3, P12 and P17 anticipated Lʌksa would not sense its location well at the Library, and refreshed the display to track the location sensing. They learned before arriving at the Library that Place and Availability were wrong. Similarly, for S4-situ, 7 participants refreshed to (impatiently) check if their status had been set to Available and/or Place as Café. In fact, because of this sluggishness, some participants attributed mis-inferences to lag.

Occasionally, participants forgot what had happened recently, *e.g.*, for S2, P07 thought he was talking to the experimenter at the time Lʌksa inferred Sound as Talking. Had he played the recorded audio of that time (Situation), he would have learned that only singing was heard. Using the played audio, P15 and P16 were able to identify guitar sounds when Sound was finally recognized correctly as Music. Hence, in combination with History, Situation explanations can help jog a user's memory of what was happening, independent of the application's inference. This helps them form Situation Justifications for the application behavior. How may we also provide Situation explanations for contexts other than Sound? For Place, perhaps by showing a photograph at the location (if one was taken at the same time). For Motion recognition, perhaps by animating an *interpreted* diagram of how the phone was moving (derived from accelerometer data).

While earlier research into intelligibility sought to prioritize providing some explanation types over others (*e.g.*, [6, 7]), along with [10], our findings suggest instead to provide a diversity of explanation types will be helpful to support different learning and troubleshooting strategies users have.

Deeper Usage of Intelligibility

Our quantitative results indicate that viewing more explanations, especially about deeper contexts can lead to deeper understanding, and more effective control suggestions for improving the application behavior. So, to promote user understanding, we need to encourage users to dig for more explanations, and to dig deeper. Perhaps, if the user starts asking questions, the application can hypothesize faults, and highlight which factors are probably causing them. These guesses could come from a knowledge base of typical faults [9], or be triggered when inferences Certainty becomes too low (*e.g.*, <80% [10]).

Intelligibility affected by Familiarity with Context Type

Also observed in [10], our participants indicated less familiarity with Sound than they did with Place, and this affected the usage and usefulness of intelligibility. Participants had fewer Control Suggestions for Sound settings than for Place, despite higher View Counts for explanations about Sound than Place (Table 4, Left). This lack of familiarity also appears to influence their perception of Explanation Helpfulness, where it was lower for S2 (Sound misinferred) than for S3 (Place misinferred), even though participants perceived the application behavior as equally poor in both scenarios. Perhaps providing easier access to Definitions and repeated exposures to the intelligibility features can promote familiarity, and allow users to gain more understanding of more novel contexts.

Intelligibility for Control

The lack of familiarity with Sound also hindered our participants' ability to provide Control suggestion to improve its inference. Only a few suggestions were made: *e.g.*, P10 suggested using her iTunes music library as a training dataset, and P09 suggested *"adjusting the levels for Periods of Silence for conversation vs.*

music." Clearly, we should provide intelligibility to facilitate control (as recommended by [7]), perhaps by just adding a Definitions-style explanation that describes control mechanisms and some consequences of adjusting them.

Constraints for Intelligibility

While the upper bounds of our participants' usage of intelligibility may give an indication of engagement, the lower bound may portend the limits to which some users may be willing to use intelligibility. Therefore, we derive some time and view constraints for intelligibility. Participants only spent about 3-10 seconds viewing each explanation, so each explanation page needs to be correctly and effectively interpreted within that short duration. Perhaps if an explanation cannot be understood within that duration, it should be split into multiple parts where the user can ask for more *on demand*. Furthermore, our quicker participants spared only about 1-3 minutes exploring explanations for each incident. This may be even shorter without the experimenter demand effect when users explore intelligibility outside of a user study. Hence, question asking should be streamlined facilitate multiple views (~20) within about 2 minutes before the user gives up.

With our scenarios, we have focused on investigating the usage of intelligibility about incidences *in-situ* and in the moment (or shortly after), or a future hypothetical situation. However, users may postpone investigating an incident until they have more time. Under those circumstances, the time constraints for using intelligibility may not be so tight, but users may need more information to remind them what happened (*e.g.*, richer Situation and History explanations).

How much Intelligibility is "Good Enough?"

We have discussed how intelligibility can be beneficial to users by improving their understanding of application inference and behavior. We have also discussed how to amplify these benefits through deeper use of intelligibility. However, is this deeper use sufficient or excessive, *i.e.*, what are the upper and lower bounds for how much intelligibility to provide? Our results suggest that users may use intelligibility at least to the depth that was provided in Lᴀksa, and therefore it is not *excessive*. To decide how much intelligibility usage will be *sufficient* for users (in terms of benefit, independent of effort required), we can consider a model of how intelligibility influences understanding, trust, and control (see Figure 6). This study covered the path from Usage to Control (in black), while [6, 7, 10] have also explored the link between Understanding and Trust. In particular, [10] has some recommendations about when and how much intelligibility to provide to help instead of harm impression (Trust). From Figure 5, Right, we can see that our participants gave at least one effective Control suggestion on average. If one change is sufficient to improve the application's performance, then this should be good enough. However, this depends on the type of suggestion (see Figure 3): changing a Rule is brittle and may not be a robust solution in the long-run; changing Behavior puts extra strain on the user's actions and habits, so they may not favor this repeatedly too; changing Settings is more robust and less tedious, but may suffer from side-effects (*e.g.*, adjusting weights in a machine learning system). Therefore, users may need to have more suggestions to be able to provide satisfactory control, which means that they should explore explanations of deeper contexts more than of the shallower application context.

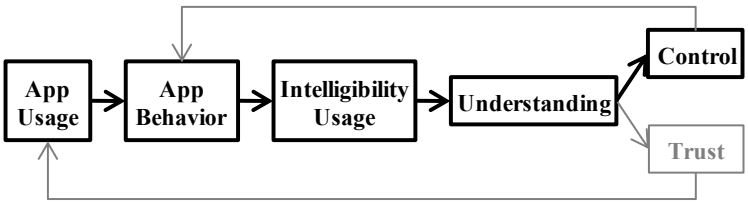


Figure 6. Model of influence indicating how the usage of Intelligibility may influence a user's ability to Control, sense of Trust, and usage of the context-aware application. Lines in grey were not explicitly explored in this study.

LIMITATIONS AND FURTHER WORK

While our quasi-field study with an interactive prototype and realistic scenarios can provide insight into how users use and benefit from intelligibility, there are limitations due to its controlled set-up and brief duration. Our study covered only a handful of situations where intelligibility is useful, but we expect more situations and even *unanticipated* ones as users use Lαĸsa in their daily lives. Furthermore, participants had only two hours to familiarize themselves with the UI, and go through four scenarios. As such, their experience only covered the *initial transient* usage of intelligibility, as novices. We expect their usage patterns and knowledge of the application and its inference to evolve as they use intelligibility over time. Therefore, for future work, we plan to deploy Lαĸsa in the field and over a few weeks to overcome the aforementioned limitations. We intend to study how prolonged use of intelligibility impacts long-term understanding and trust of context-awareness.

CONCLUSION

We have presented a quasi-field study where we measured how participants naturalistically used an intelligible context-aware application in scenarios representing real-world, "everyday" situations. We investigated how that usage affects their understanding of the application behavior. The application was an iteration over the Lαĸsa prototype with more streamlined and usable intelligibility features. We found that viewing more explanations, especially more about deeper contexts can further improve user understanding of application inference. We provided implications for promoting more effective intelligibility usage, time constraints within which users are willing to view intelligibility, and discuss how much intelligibility should be provided to sufficiently improve user understanding of context-aware applications.

ACKNOWLEDGEMENTS

This work was funded by the National Science Foundation under grant 0746428, Intel Research, and the Agency for Science Technology And Research, Singapore. We thank for their feedback: Aniket Kittur, Denzil Ferreira, Christian Köhler, Tawanna Dillahunt, Bertha Lam, Matthew Lee, Ian Li, SeungJun Kim, Dezhong Yao, Minkyung Lee, Ruogu Kang, Scott Davidoff, and Brian Ziebart.

REFERENCES

1. Bellotti, V. & Edwards, W.K. (2001). Intelligibility and Accountability: Human Considerations in Context-Aware Systems, *Human-Computer Interaction*, 16(2-4): 193-212.
2. Cheverst, K. *et al.* (2005). Exploring issues of user model transparency and proactive behavior in an office environment control system. *UMUAI 05*, 15(3-4), 235-273.
3. Dey, A.K., Abowd, G.D. & Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *HCI Journal*, 16(2–4): 97–166.
4. Hilbert, D.M., & Redmiles, D.F. (2000). Extracting usability information from user interface events. ACM *Computing Survey* 32(4), 384-421.
5. Kern, N., & Schiele, B. (2006). Towards Personalized Mobile Interruptibility Estimation. *International Workshop on Location and Context-Awareness*, 134-150.
6. Lim, B.Y. *et al*. (2009). Why and why not explanations improve the intelligibility of context-aware intelligent systems. *CHI 09*, 2119-2128.
7. Lim, B.Y., & Dey, A.K. (2009). Assessing Demand for Intelligibility in Context-Aware Applications. *Ubicomp 09*, 195-204.
8. Lim, B.Y., & Dey, A.K. (2010). Toolkit to Support Intelligibility in Context-Aware Applications. *Ubicomp 10*, 13-22.
9. Lim, B.Y., & Dey, A.K. (2011). Design of an Intelligible Mobile Context-Aware Application. *MobileHCI 11*, to appear.
10. Lim, B.Y., & Dey, A.K. (2011). Investigating Intelligibility for Uncertain Context-Aware Applications. *Ubicomp 11*, to appear.

11. Lu, H. *et al*. (2009). SoundSense: scalable sound sensing for people-centric applications on mobile phones. *MobiSys 09*, 165-178.
12. Milewski, A.E., & Smith, T.M. (2000). Providing Presence Cues to Telephone Users. *CSCW 00*, 89-96.
13. Muir, B. (1994). Trust in automation: Part i. theoretical issues in the study of trust and human intervention in automated systems. *Ergonomics*, 37(11): 1905–1922.
14. Oulasvirta, A. (2005). Grounding the Innovation of Future Technologies. *Human Technology* 1 (1):58-75.
15. Rosenthal, S., Dey, A.K., Veloso, M. (2011). Using Decision-Theoretic Experience Sampling to Build Personalized Mobile Phone Interruption Models. *Pervasive 11*, 170-187.
16. Roto, V. *et al.* (2004). Examining Mobile Phone Use in the Wild with Quasi-Experimentation. Helsinky Institute for Information Technology (HIIT), Technical Report, 1, 2004.
17. Rukzio, E. *et al*. (2006). Visualization of uncertainty in context aware mobile applications. *MobileHCI 06*, 247-250.
18. Tullio, J. *et al.* (2007). How it works: A field study of non-technical users interacting with an intelligent system. *CHI 07*, 31-40.
19. Vermeulen, J. *et al*. (2010). PervasiveCrystal: Asking and Answering Why and Why Not Questions about Pervasive Computing Applications. *IE 10*, 271-276.
20. Weiser, M. & Brown, J.S. (1997). The coming age of calm technology. *Beyond Calculation: the Next Fifty Years*, 75-85.
21. Weka for Android. https://github.com/rjmarsan/Weka-for-Android. Retrieved 26th August 2011.
22. Welbourne, E., Balazinska, M., Borriello, G., Fogarty, J. (2010). Specification and Verification of Complex Location Events. *Pervasive 10*, 57-75.