Assessing Demand for Intelligibility in Context-Aware Applications

Brian Y. Lim, Anind K. Dey
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213, USA
{byl, anind}@cs.cmu.edu

ABSTRACT

Intelligibility can help expose the inner workings and inputs of context-aware applications that tend to be opaque to users due to their implicit sensing and actions. However, users may not be interested in all the information that the applications can produce. Using scenarios of four real-world applications that span the design space of context-aware computing, we conducted two experiments to discover what information users are interested in. In the first experiment, we elicit types of information demands that users have and under what moderating circumstances they have them. In the second experiment, we verify the findings by soliciting users about which types they would want to know and establish whether receiving such information would satisfy them. We discuss why users demand certain types of information, and provide design implications on how to provide different intelligibility types to make context-aware applications intelligible and acceptable to users.

Author Keywords

Context-aware, explanations, intelligibility, satisfaction

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms: Human Factors.

INTRODUCTION

As systems and devices get closer to realizing a vision of ubiquitous and calm computing, *context-aware applications* will become more common. Context-aware applications use *context* – information about the state of people, places, and objects relevant to users and their activities [9] — to make inferences as to how to adapt their behavior. They typically gather context from a combination of sensors and complex rules, and take action automatically without explicit input from users. This lack of transparency along with the occasional fallibility of context-aware applications can hinder users' attempts to make sense of these applications [6]. Such a lack of application *intelligibility* can lead users to mistrust the system, misuse it, or abandon it altogether [22]. Specifically, users of context-aware applications can find this lack of intelligibility frustrating [5].

As a remedy, Bellotti and Edwards state that context-aware applications must be intelligible: being able to "represent to their users *what* they know, *how* they know it, and what they are *doing* about it." [6]. One way to support intelligibility is through automatically generating explanations of application behavior. This approach has been employed in several domains including decision making [11], recommender systems [17], end-user debugging [18, 19], and systems that model user profiles [7], with the goal of increasing user trust and acceptance of these systems. While studies of these systems include explanations in working systems, little work has been done to compare the impact of different types of explanations or in the domain of context-aware computing [20].

In this paper, we categorize types of explanations to support intelligibility in terms of questions users may ask of context-aware applications. In earlier work, we found that some types of explanation were more effective than others in improving users' understanding and trust of a context-aware intelligent system [20]. However, it is not clear what information users actually want to know and will ask about. Our contribution is to explore and assess user demand for intelligibility: which types of questions users want answered, and how answering them improves user satisfaction of context-aware applications. User satisfaction is obviously crucial for adoption and acceptance of such technologies.

The paper is organized as follows: we discuss how supporting intelligibility by providing explanations that users want, has the potential to increase user satisfaction and thus acceptance of context-aware applications. We then describe our experimental design that uses surveys and scenarios to expose users to a range of experiences with context-aware applications. We present two experiments that investigate what types of information users want. In the first experiment, we elicit the types of information users are interested in and under what moderating circumstances. In the second experiment, we validate our findings by presenting users with 11 information types as intelligibility features in a controlled study and measure their impact on user satisfaction. We end with a discussion of why users of context-aware systems demand certain types of information in different situation, and provide design recommendations for providing different information types to make context-aware systems intelligible and acceptable to users.

UbiComp 2009, Sep 30 – Oct 3, 2009, Orlando, Florida, USA.
Copyright 2009 ACM 978-1-60558-431-7/09/09...$10.00.

ASSESSING DEMAND FOR INTELLIGIBILITY IN CONTEXT-AWARE APPLICATIONS

To make context-aware applications intelligible so that they can expose their inner functions to the end-user, much research has looked into how to generate explanations from the underlying application models and deliver them to users (*e.g.*, [7, 18, 19, 20]). However, users may not be receptive to these explanations, especially when they end up using the applications in ways for which they were not designed [24], and when those explanations do not adapt to varying situations of use. Thus it is important to explore information demand from the user's perspective lest effort is wasted in implementing explanations that would see little use.

Researchers have explored what users want to know in other domains. McGuinness and colleagues [14, 21] have identified information need factors that influence the level of trust in adaptive agents. They used interviews to identify explanation requirements and rank question types according to their helpfulness. Gregor and Benbasat's [16] meta-review investigates explanation types that users of knowledge-based systems (KBS) would like to have. While adaptive agents and KBS are similar to context-aware applications (which may also use agents or knowledge bases and rules), they are work-oriented, while context-aware applications are targeted for everyday use, for many more situations and a wider range of users, and under more situations [2]. Thus we need to explore how these different requirements would lead to different intelligibility needs.

Hypotheses and Approach

We hypothesize that there are different types of information users are interested in, for different context-aware applications, and different situations. Since people ask information seeking questions due to cognitive disequilibrium [15] and to correct knowledge deficits [27], we believe that satisfying these information demands through intelligibility can lead to better satisfaction when using these applications and improved adoption and acceptance. In order to elicit the information demands users have for context-aware applications under various situations, we conducted a study of the demand for explanations and different types of information in several scenarios users may find themselves in as they use context-aware applications. Using described scenarios instead of actual field deployments allows us to quickly and more effectively study and understand the impact of different information on intelligibility and satisfaction, without having to implement and deploy a variety of applications, any of which could fail for reasons independent of our main focus. Next we describe four applications we use to focus our scenarios. For each application, the scenarios intentionally span a range of incorrect, appropriate and unexpected or anomalous, but not necessarily wrong behavior, to probe directly at the issues of intelligibility and satisfaction.

SETUP: SCENARIOS OF FOUR CONTEXT-AWARE APPLICATIONS

To investigate the demand for intelligibility in the space of context-aware applications, we selected four prototypical context-aware applications: (i) a desktop interruptibility management application (an Instant Messenger plugin), (ii) a remote person monitoring peripheral display (Digital Family Portrait), (iii) a context-aware reminder application (CybreMinder), and (iv) a mobile context-aware tour guide (CyberGuide). All applications in this study behave according to models of learned decision trees.

Instant Messenger Auto-Notification

We designed the instant messenger (IM) auto-notification plugin based on recent [4] work on a predictive model to determine how long a buddy would take to respond to a message. Our application uses the responsiveness prediction to determine the subject's interruptibility [12], and either *forwards* or *suppresses* incoming IM messages. We developed four main scenarios for this application where the subject is in various states of busyness:

1. Rushing to reach an imminent deadline,
2. Taking a break and surfing the Internet,
3. Reading a work-related book, and
4. Returning from a protracted informal meeting.

For each scenario, the user receives an IM message from

- A colleague regarding critical work, or
- A friend regarding a fun video.

There are 16 scenarios (4 main x 2 messages x 2 actions).

Remote Monitoring: Digital Family Portrait

The Digital Family Portrait [23] leverages a picture frame to present the current status of an elderly family member as he or she goes through daily life living independently in her home, to remote loved ones. Our rendition of the Digital Family Portrait is based on a decision tree model which we define as several small subtrees, each addressing groups of scenarios. We present a subset of what the sensors on the elder's body and in the home are described as detecting:

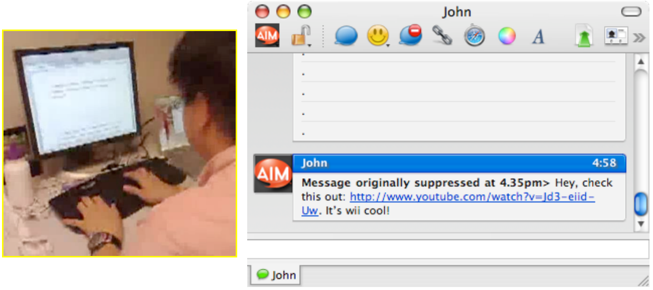
1. Whether the family member has fallen,
   Whether there is a fire;
2. How many times the toilet has been used recently,
   Whether the usage frequency is anomalous,
   Whether the system thinks this could be a symptom of incontinence;
3. Whether the family member is watching TV,
   Whether the family member is sleeping
4. Whether the family member's house is vacant,
   Whether there is an intruder.

For this application, there are a total of 13 scenarios.

Reminders: CybreMinder

CybreMinder [8] is a context-aware reminder application that considers combinations of contexts, such as location, time, and collocation, to trigger reminders. It is based on several personal and environmental sensors, and triggers reminders based on the satisfaction of one of several rules (modeled as a decision tree). We developed scenarios that would relate to three types of reminders (mentioned in [8]):

1. Reminder to discuss an important issue when the user and a colleague serendipitously meet (collocation trigger);



Message originally suppressed at 4.35pm> Hey, check this out: http://www.youtube.com/watch?v=Jd3-eiid-Uw. It's wii cool!

Figure 1: (Left) Screen capture of a five-second video clip for the IM Auto-Notification application survey showing the user rushing to meet a deadline. (Right) Screenshot of a non-work IM message that had been suppressed, and delivered later.

2. Reminder to take the umbrella when it is forecasted to rain and the user is approaching the front door (location and information trigger); and,
3. Reminder to discuss party planning with a friend when the user and the friend are free, and the user is at the office (complex trigger).

We developed 13 scenarios based on these three reminders.

Tour Guide: CyberGuide

CyberGuide [1] is a mobile context-aware tour guide that uses context to recommend attractions to a user. Our rendition of CyberGuide considers the contexts of location (where the user is currently located), keywords (that the user has recently used), and navigation information like traffic. We use three settings for a user with a keen interest in museums and dinosaurs visiting an unfamiliar city:

1. Walking by museum with a dinosaur exhibit;
2. Having a conversation with a friend talking about museums and dinosaurs; and,
3. Meeting a friend at his home with the application recommending a route to the destination.

We developed 12 scenarios based on these three settings

EXPERIMENT 1: ASSESSING DEMAND FOR INFORMATION TYPES

Based on these four applications and the scenarios we developed, we conducted our first experiment on the intelligibility of context-aware applications to investigate what questions users want to ask in the various scenarios.

Method

We created one survey for each of the four prototypical context-aware applications. At the beginning of each survey, the respective context-aware application is described to the participant. Its functionalities are described, but not explained or elaborated on (*e.g.,* participants are not told where it gets its information from). Depending on the application survey, participants are shown 3-4 scenarios, described from a first-person perspective that places participants in certain circumstances (*e.g.*, rushing to complete a report due in half an hour for the IM application). The scenarios are represented by short 5-second video clips (see Figure 1 left) and short textual descriptions about what is happening. After a scenario is presented, participants are shown 2-5 (depending on the application and scenario) instances of the scenario, one at a time, with different application responses, represented as screenshots along with text (*e.g.*, see Figure 1 right), where the behaviors may be appropriate, strange, or incorrect. For each application response, participants are reminded about the scenario and can replay the video, if necessary.

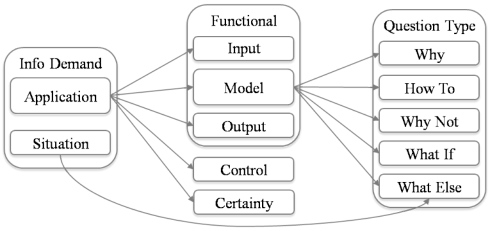


Figure 2: Hierarchical representation of intelligibility types that users want to find out about.

To mitigate order effects, the order of scenarios was randomized for each participant. For each scenario, we posed several questions (see Table 1) to ascertain what the participant thought and felt about the application response, and what information they would want to know, if any, for the situation. Except for a question on application satisfaction with a 7-point Likert scale response, the rest of the questions were free text response.

We recruited 250 participants (47% female; ages 18 to 61, M=29.5) from Amazon Mechanical Turk, and paid them $4 for completing a survey. Participants were divided among the four applications surveys, and survey analysis was conducted between-subject.

Coding Analysis: Intelligibility Types and Moderators

We coded the participant responses to the free response questions to determine whether participants wanted more information regarding the application or situation, and what type of information they wanted. Using the open coding method of grounded theory [26], and drawing from question types employed previously [3, 20], we derived a set of information types that users are interested in for context-aware applications. In the rest of the paper, we shall refer to these types as *intelligibility* types. Figure 2 presents them in terms of a hierarchy and Table 2 shows the coding scheme used. First, depending on the situation, participants may or may not have any *information demand*. If they want more information, they may want to know more about how the application works, in terms of functional *input*, *output* and conceptual *model*. A conceptual model describes the decision and action processes

Measure	Survey Question
Application Satisfaction	I am satisfied with the application response (7-point Likert scale)
Action	What will you do?
User Feeling	How do you feel about what the application did?
Information Demand	What information or knowledge would you like to know about what the application did or why it behaved this way?

Table 1: Questions posed to participants for each application response scenario to find out what they think the application response, what they think is happening, and their information demand for each scenario.

performed by the application to use the inputs in producing output. Regarding the conceptual model, the participant may ask for information that can be represented in the following questions:

1. *Why* did the application do X?
2. *How* (under what condition) does it do Y?
3. *Why* did it *not* do Y?
4. *What (else)* is it doing?
5. *What if* there is a change in conditions, what would happen?

Question 1 (*why*) and 3 (*why not*) seek situation-specific information, while question 2 (*how*) seeks more comprehensive information of how the output can be obtained. Although context-aware applications that sense and act implicitly may elicit questions on what the application is doing, question 4 (*what else*) does not ask this. Instead, participants were interested in knowing what else the application was doing (*e.g.*, whether the Digital Family Portrait had contacted emergency services after detecting the elderly family member had fallen). Regarding non-functional information about the application, users may want to know how certain the application is of its actions, decisions, and inferences (*certainty*). In agreement with past researchers [5, 6, 10], users also want to be able to *control* the application (*e.g.*, change settings for reminders), and want information on how to control it. In addition to more information about the application, the user may also want to know more about the current *situation*. Regarding the situation, participants expressed their questions mostly in terms of *what else*, and sometimes as *why* or *why not*.

We also used the survey responses to derive themes describing circumstances that *moderate demand* for these intelligibility types:

- **Application Satisfaction.** (7-pt Likert scale) How satisfied the user is with the application behavior.
- **Impression.** Coded response of whether the user has a *positive*, *neutral*, or *negative* impression of the application, given the situation.
- **Trust / Reliance.** Coded response of whether the user is

Theme	κ	Values / Description		Example Participant Text
Impression	.91	–	Negative (Useless / Dissatisfied / Irritated / Frustrated)	“Angry” “Bad” “I'm pissed off it could have cost me my job” “I feel cheated” “I'm disappointed in the application”
		0	Neutral / equivocal	“It's ok.” “fine”
		+	Positive (Useful / Satisfied)	“Good. Positive.” “I love it, works like a charm.”
Trust of Device	.91	0	Abandon it, Complain, Lost Trust, Doubt it, Dissatisfied / believe it is wrong	“I wouldn't be paying attention to it anymore” “Something must be wrong with one of the sensors.” “It may have misinterpreted an action.”
		1	Satisfied / Believe / Trust	“The application performed properly.” “I'm satisfied.” “It's okay, it was my fault.”
Information Demand	.84	0	None / Not necessary	-
		1	Too much info already / Overwhelmed	“It is more than I would want to know.”
		2	Yes / Not enough info/details	*(See below)*
Domain of Explanation	.95	0	None	-
		1	Application, Device, Sensors (includes logic)	*(See below)*
		2	Situational / Event	“My status and Johnny's priority.” “How often the person moved, or what the rate of respiration was.” “Our locations and distance.”
Input	.95	Sensor thresholds, ranges, sensitivity, limitations, capabilities, coverage, etc		“I would like to know what triggers the alarm” “Where does it get its forecasting information? How did it know I was leaving rather than entering my house?”
Model	.97	Application conceptual model / Criteria / Heuristic / Rules / Logic / How does it know / how it works		*(See below)*
Output	1*	Options / Alternatives that the system could produce. *Distinguish from What Else*		“I would like to know fully what type of things that it would be able to detect *[recognition output]*” “I would want to know if there was a quicker route.”
Why	1	*Why* did the application do X?		“Why did it let *[the message]* through?” “How does it know it is going to rain?”
Why Not	.96	*Why* did it *not* do Y?		“I would like to know why the application did not filter out this message.” “Why application failed to sense that I was free in the office and failed to trigger.”
What Else	.95	*What (else)* is it doing?		“What other message are in the queue” “My location, weather, and time.” “How I could look up more specific details about the event.”
How	.97	*How* (under what condition) does it do Y?		“I would be interested in knowing how it decides how long to suppress messages.”
What If	1*	*What if* there is a change in conditions, what would happen?		“I'd like to know in what will application do in certain situations”
Certainty	1*	Application confidence of its actions / decisions		“How can I be sure it is accurate?” “I'd like to know how accurate the chosen forecasting system was.”
Control	1*	How to change settings / thresholds		“I would want to know if I could put a time limit on it.” “I would like to know how to make it send the correct reminder.”

Table 2: Coding scheme for experiment 1. The first two themes indicate participants’ thoughts on the scenario, and the remaining themes indicate their information needs. Most of the participant text responses were coded by one coder, with a 10% random sample of responses coded by a second coder. Inter-coder reliabilities (κ) for each theme are indicated.
** denotes high apparent reliability due to low occurrence of coded measure (i.e., too few affirmative counts).*

satisfied (believes application, trusts it) or *dissatisfied* (doubtful, lost trust, in disbelief, found fault, will abandon) with the application.

These measures were found to be correlated and so we combined them into a summed and reversed measure of application **Inappropriateness** (α=.68). We split the scenarios into 2 groups: those with a high or low Inappropriateness score (using a Tukey pair-wise test; p<.001).

Based on the application functionality in various scenarios, we developed more moderators. For example, because CybreMinder supports goals of pre-planned tasks, while the other 3 applications do not, scenarios can be separated into whether they are goal-supportive or not. The additions are:

- **Criticality.** Whether the situation presented is critical. Situations involving accidents or medical concerns with the Digital Family Portrait and work-related urgency for the IM Auto-Notification were considered highly critical. Due to the profound influence of the high criticality of the fall and incontinence scenarios in the Digital Family Portrait survey, these scenarios were excluded from consideration of the other moderators.
- **Goal-Supportive.** Whether the situation is motivated by a goal the user has (CybreMinder scenarios only).
- **Recommendation.** Whether the application is recommending information for the user to follow or ignore (CyberGuide scenarios only).
- **Externalities.** Whether the application is perceived to have high external dependencies (*e.g.*, getting weather information from a weather radio station) *vs.* being perceived as "self-contained." CybreMinder and CyberGuide had perceived high external dependencies.

	Average (%)	Inappropriateness	Criticality	Goal-Supportive	Recommendation	Externalities
Information	72.5	↑↑			↑	↑↑
Application	59.8	↑	↓↓		↑↑	↑↑
Situation	12.0		↑↑	↑*	↓	
Input	19.8		↓		↑↑	↑↑
Model	33.6	↑↑		↑		
Output	2.9			↓	↑↑	
Why	19.0	↑				
Why Not	8.2	↑↑		↑↑		↑↑
How	13.4					
What If	0.5					
What Else	6.4					
Certainty	1.7					
Control	10.0	↑				

Table 3: Results of experiment 1. The left column shows the percentage of participants who had a demand for various intelligibility types. The right columns show the effect size of whether higher moderator rating values (of each column) leads to increased (up arrows) or decreased demand. All results indicate Bonferroni-corrected (n=78) significant differences (p<.01; * denotes p<.05). Cohen's *d* is reported to determine the size of differences rather than just whether the differences are significant. Single arrow indicates small effect size (.2<|d|<=.3), double arrows indicates medium effect size (.3<|d|≤.8).

How to read: *e.g.*, 72.5% of participants demand information, in general; participants demand more information about *why not* when the application behavior is more Inappropriate.

Results: User Demand of Information Types

Results are shown in Table 3, describing the overall demand for various types of information, and how the demand changes due to moderating circumstances. Discussion is deferred until after we present the results of experiment 2.

EXPERIMENT 2: ASSESSING DEMAND FOR EXPLANATION TYPES

While experiment 1 sought to elicit the types of information users wanted for explanations of context-aware applications under various moderating circumstances, experiment 2 solicits its user demand for the types of information through explicit suggestion of questions and provision of explanations and studies the impact of meeting this demand on user satisfaction. For each of the intelligibility types identified in experiment 1, we wrote corresponding questions and explanations for each application situation response of experiment 1. We added one more intelligibility type in response to demand for more information about the application conceptual model. This "Visualization" explanation type provides a diagrammatic representation of the underlying decision tree. Table 4 shows examples of questions and explanations. We prepared 11 intelligibility types for each of the 54 scenarios, for a total of 594 explanations; some are repeated for similar scenarios.

We hypothesize that: (i) when asked specifically about whether they want an intelligibility type (heretofore called *solicited information demand*), users should reflect the same demands (*elicited information demand*) as that of experiment 1; and providing explanations for demanded intelligibility type will (ii) increase application satisfaction, and (iii) increase user rating of that intelligibility type.

Method

We reused the applications and scenarios from experiment 1, with one application per survey. Experiment 2 is designed as a between-subject study for the intelligibility types (11 conditions plus a None condition). Participants are assigned to a version of the survey with only one type of intelligibility information provided (including None). To mitigate order effects, four versions of each survey were deployed with a different random ordering of scenarios. On the survey, participants use a 7-point Likert scale question to rate their satisfaction with the application. In the intelligibility type conditions, participants were provided with a corresponding intelligibility question and explanation. We posed additional queries regarding how important it was to get the question answered, how much they are satisfied with the explanation and how useful they found it. We recruited 610 participants (42% female; ages 18 to 61; M=28.9) from Amazon Mechanical Turk, and evenly distributed them across the 12 conditions. Participants were paid $2.

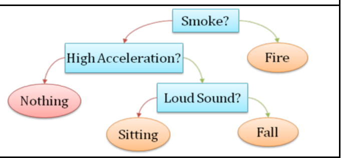
Intelligibility Type	Sample Question	Sample Explanation
Input	What does the system use to sense accidents?	The system uses an accelerometer worn by the elderly family member, speakers around the home, and smoke detectors.
Output	What accidents can the system sense?	The system can sense the following accidents: Falls and Fire/Smoke.
Why	Why did the system report a fall?	The system reported a fall because there was a high acceleration from the accelerometer worn by the family member, and there was a loud sound.
How	How does the system distinguish a between a falling object and person?	The system did not report a fire, because the smoke detector did not set off.
Why Not	Why did the system not report a fire?	The system detects a fall through high acceleration detected from the accelerometer worn by the elderly family member, and a loud noise. The system detects a fall of an object through a loud noise, but no high acceleration from the accelerometer.
What If	If an object falls, would the system report a fall?	If an object falls, but the accelerometer worn by the family member does not report a high acceleration, the system would not report a fall.
What Else (Application)	Did the system alert emergency services of the accident?	The system has not alerted emergency services and is pending your approval.
Conceptual Tree Model Visualization	What is the overall model of how the system works?	The following diagram describes a simplified view of the conceptual model of how the system works. It works by tracing a decision tree and taking branches at decision points to arrive at a conclusion. Red arrows indicate false conditions, and green indicates true. For example, if smoke is detected, then the system concludes there is a fire.  Smoke? High Acceleration? Fire Nothing Loud Sound? Sitting Fall
Certainty	How certain is the system of this report?	The system is 90% certain of this report.
Control	How can I change settings to control the sensitivity for reports?	Some settings can be changed through a control panel accessed via the menu: Options > Settings.
Situation (What Else)	What was the family member doing before the accident?	The family member was preparing dinner in the kitchen.

Table 4: Sample questions and explanations of various intelligibility types participants received for the Digital Family Portrait.

Results Analysis

We report three metrics in experiment 2: (i) *solicited intelligibility demand*, (ii) *satisfaction* of the application with and without intelligibility, and (iii) user rating of the *usefulness of intelligibility*. Table 5 summarizes the results of experiment 2 and compares them with experiment 1. Results are reported for intelligibility types across all scenarios, and by moderating circumstances (high *vs.* low rating groups) as described in experiment 1.

Solicited Intelligibility Demand (siD)

Participant solicited intelligibility demand is measured from responses to the 7-point Likert scale question on how important it is to receive an answer to the supplied intelligibility type question. All described differences are significant (p<.05) using a Tukey pair-wise comparison test. As with experiment 1, the moderating effects are measured by the effect size between low and high moderator rating groups.

Difference in Application Satisfaction (ΔaS)

To calculate relative application satisfaction from None, for each scenario, we subtracted the means of satisfaction of the

	Average				Inappropriateness				Criticality				Goal-Supportive				Recommendation				Externalities			
	Exp 1	Exp 2			Exp 1	Exp 2			Exp 1	Exp 2			Exp 1	Exp 2			Exp 1	Exp 2			Exp 1	Exp 2		
	eiD (%)	*siD*	Δ*aS*	*iuR*	*eiD*	*siD*	Δ*aS*	*iuR*	*eiD*	*siD*	Δ*aS*	*iuR*	*eiD*	*siD*	Δ*aS*	*iuR*	*eiD*	*siD*	Δ*aS*	*iuR*	*eiD*	*siD*	Δ*aS*	*iuR*
Information	72.5	5.18	0.11	4.91	↑↑		0.47	-0.47		0.81		0.45					↑				↑↑			
Application	59.8	5.22	0.09	4.9	↑		0.45	-0.48	↓↓	0.79		0.44					↑↑				↑↑			
Situation	12.0	4.82	0.28	4.92			0.76		↑↑	1.15		0.62	↑*		-0.71	-0.75	↓							-0.52
Input	19.8	5.61	0.15	5.17			0.44*	-0.45	↓	0.50*				0.61			↑↑	-0.66			↑↑			
Model	33.6	5.04	0.01	4.7	↑↑		0.46	-0.52		0.83		0.41	↑	0.42				-0.45		-0.42				
Output	2.9	5.62	0.06	5.11			0.59			0.75		0.53	↓			-0.56	↑↑							-0.48*
Why	19.0	5.27	-0.36	4.32	↑			-0.65		1.21														0.59
Why Not	8.2	4.57	-0.01	4.42	↑↑		0.68			0.91			↑↑	0.95		0.74		-0.68			↑↑			0.44
How	13.4	5.20	0.22	5.18				-0.48				0.54		0.73				-0.67	-0.80	-1.11			-0.63	-0.75
What If	0.5	5.15	0.11	4.64			0.49			0.97								-1.33		-1.16		-0.84		-0.88
What Else	6.4	5.09	0.04	4.83				-0.83	↑	0.81														
'isualization		5.29	0.29	4.71		-0.37	0.53	-0.56		0.84		0.88			-0.62	-0.83			0.54					
Certainty	1.7	5.25	0.33	5.13				-0.66		0.99		0.42*		0.60					-0.49*	-0.51*		0.62		-0.43
Control	10.0	5.37	0.23	5.42	↑		0.79			0.54*		0.53				-0.80								-0.60

Table 5: Results of experiment 2 and experiment 1. Experiment 1 results repeated for comparison. *eiD*: elicited information demand from experiment 1. *siD*: solicited information demand from experiment 2. Δ*aS*: difference in application satisfaction of providing intelligibility type from none. *iuR*: intelligibility usefulness rating of explanation received. For experiment 2, left columns under Average are the mean values of the Likert scale for *siD* and *iuR*, and Δ*aS*, the difference in means between intelligibility-provided (various types) and non-intelligible. The right columns are differences in means or differences between high and low moderator rating groups. All experiment 2 results of each measure are Bonferroni corrected (n=84) and significant (p<.01, * denotes p<.05).

How to Read: e.g. for Intelligibility type *Situation*, 12% of participants in experiment 1 were interested in knowing about the situation; in experiment 2, participants wanted to know about it (M=4.82, above neutral); they rated it well, in general (M=4.92) and had a significant improvement in application satisfaction (ΔM=0.28). When considering scenarios according to Criticality, participants want more information about the situation (both elicited and solicited, both medium effects), experienced more satisfaction after receiving explanations for more critical scenarios (p<.05), and rated situation explanations better for those scenarios (small effect).

None condition from each response in the 11 Intelligibility type conditions. Across moderating circumstances, we calculate the effect size between the high and low moderator groups. If the effect size is small or medium, we report the *difference* in scale value mean along with the p value of a t-test between the groups.

Intelligibility Usefulness Rating (iuR)

We derived an intelligibility usefulness rating based on the mean of responses regarding explanation satisfaction and explanation usefulness (Cronbach's α=.84). We used the same analysis as described for *siD*.

DISCUSSION

We discuss the meaning of the results from both experiments in terms of demand for intelligibility and what they imply for design. Among the four context-aware applications we investigated and their various scenarios, we have determined average demands for various intelligibility types. However, these demands change depending on circumstance and function of the application. Participant awareness to seek certain intelligibility types also increases their demand for these types. We also found that explanations should be carefully tailored to be effective, otherwise, they may be more detrimental than helpful.

Participants generally want information about the *application*, and designers can anticipate this, but they also moderately want information about the *situation* in which the application acts. When asking about the application, participants tend to focus on the application *model* rather than the *inputs* or *outputs*. Of the application model, they care most about *why* the application behaved as it did in specific situations, and *how* it works in general and *how* to produce certain actions or decisions. Participants indicated a strong demand for *visualizations* (with significant positive change in application satisfaction, and intelligibility explanation rating). This may be because participants prefer graphical to textual explanations, so visualization-based explanations may be a better means to deliver explanations. As in [20], *why not* explanations had a significant but limited demand compared to why. As indicated by many researchers (e.g., [5, 6, 10]), our participants were interested in knowing how to *control* applications. We found that users were more satisfied with receiving this intelligibility type even though we only told them where they could find a control panel. This echoes findings in [14] of how users adopt a "trust but verify" approach, and just need to know they can override the settings, but not necessarily do so.

Demand for Intelligibility Varies with Circumstances

The *why not* intelligibility type is particularly effective for Inappropriate circumstances and Goal-Supportive functions. Participants would tend to ask why not when encountering an inappropriate behavior, especially if it deviates from their goals. *Output* information is desired more for Recommenders and *input* information for applications with high Externalities. Table 5 indicates other intelligibility types that vary by circumstances.

High Intelligibility Demand for Critical Circumstances

Criticality is a particularly significant moderating circumstance as for highly critical scenarios, participants want as much information about any intelligibility type (especially about the *situation* and *what else*) as they can get if they are aware of its availability. However, it is hard to satisfy participants with any intelligibility types in highly critical scenarios, possibly due to the stress during the critical period, and preoccupation with the primary problem.

Awareness of Intelligibility Types Changes Demand

Just as in [21], we found that awareness of the existence of intelligibility types plays an important factor where participants recognize the value of the various types and subsequently had greater demand for them, resulting in higher application satisfaction; this created higher demand and benefit than would have been predicted from experiment 1. Though not in demand when elicited, *input*, *output*, *what if*, *what else*, and *certainty* became particularly desired when participants were informed or reminded about them.

Valley of Expectation

The elicited and solicited information demand measures indicate which intelligibility types are desired and under what circumstances, but providing explanations matching these types in the corresponding circumstance may lead to poorer intelligibility usefulness ratings instead. This is particularly evident for Inappropriateness circumstances, but absent for highly Critical ones. This suggests that when participants have a higher demand for intelligibility, they also expect better explanations, and may rate explanations worse. However, when they become more desperate for information (as in cases of high criticality), they readily accept the explanations, and rate them more favorably. We refer to this drop then rise in expectation of explanation quality with respect to information demand as the "valley of expectation" of intelligibility demand.

Satisfaction vs. Understanding

This work has explored the intelligibility types that users demand and that may satisfy them. We compare and contrast our findings to our earlier work in exploring how to improve user understanding and trust of a context-aware intelligent system [20]. In the earlier work, we found that *why* and *why not* explanations were most effective, while interactive explanation types (*how to* and *what if*), were less useful due to the difficulty of using them. In this study, we provide participants with various intelligibility types without involving any user interaction, so we avoid the confound of difficult user interfaces and, in fact, find that *how* (*to*) and *what if* explanations should not be neglected. We also found that even though *why* and *why not* explanations are intuitive and particularly good at improving understanding, users have higher expectations on the informational quality of such explanations, and may be biased against appreciating them as much.

DESIGN RECOMMENDATIONS

We present some recommendations to developers of context-aware applications about which intelligibility types to provide and under what circumstances. Developers can identify which moderators and situations apply for their applications and then use the suggested recommendations. Implementing all the types of intelligibility we investigated is excessive and may even be detrimental. We present the different types and offer advice about when and how they should be implemented (summary in Table 6 and 7).

Application. When users want information, they will tend to want application-centric information, so, in general, more effort should be concentrated on providing information about the application's workings and mechanism than on the situation. Explanations about the application should also be provided for applications that are at risk of being easily abandoned (low trust), and when the application is uncertain (high risk of Inappropriateness) of its action.

Situation. Though users are less interested about what else is happening when the application responds, there is moderate interest in increasing their real-world situational awareness (*what else*). This would involve making applications sense more related events and contexts (*e.g.*, for a monitoring application, what is the historical trace of events before an anomaly) than their primary function, and to reveal such knowledge to the user. This is more important in cases when the application acts in highly critical situations.

Input. Users may only have a moderate interest in knowing more about the application's *input* sources or sensor readings, but if they perceive the application is heavily dependent on external sources (high externality), they may want to know more about the inputs.

Output. In typical application use, users are not interested in the output alternatives. However, they may suspect that the application is capable of more action than it is exhibiting. Providing intelligibility explaining the *output* (action) capabilities of the application is particularly important for applications that make recommendations (such as tour guides), especially when users want to seek better options. This should be provided automatically during early stages of usage to improve users' awareness.

Model. Most of the questions users want to ask are about an application's conceptual *model*. This is elaborated with the following intelligibility types.

Why. Answering *why* questions is an essential intelligibility requirement as such questions are very common. However, users may also have high expectations that the *why* explanations are very informative and a simple reason trace may not be sufficient to fully satisfy the users' enquiries. Visualizations could be used to augment a trace.

Why Not. Such explanations are good for high risk (high chance of inappropriateness) circumstances and goal-supportive functions. However, we caution against implementing it in all types of context-aware applications because generating these explanations for all alternative possibilities may be non-trivial.

How. Users are somewhat interested in knowing how the application arrives at its outcomes, and particularly like such explanations. However, *how* explanations can get cumbersome to produce for applications with complex or learned logic. Users may have to use an interactive facility to specify the constraints in which to obtain an action [20].

What If. This explanation type also involves user interaction in specifying input conditions and the application simulating what would happen. We recommend *what if* explanations for non-recommenders and more self-contained applications, for which users indicated strongest demand.

What Else. The *what else* explanation provides information closely related to the *situation* explanation. The latter provides situational awareness, while the former provides more information about what the application has done. Unsolicited demand for *what else* information is low, but becomes significant when participants are aware of its availability. This indicates an intrinsic need for this type of explanation. *What else* explanations are also important in critical situations when users hope that the application is doing more to remedy or handle the critical situation.

Visualization. Given the general demand and effectiveness for this intelligibility type, we recommend providing visualizations to augment explanations.

Certainty. Providing *certainty* information is particularly important for applications that are goal-supportive where users want to know how certain the application is in its decision or action, and for applications that rely heavily on external sources and sensors. *Certainty* values that we provided for this study were over 90%, but we suspect that if low *certainty* accuracies are reported to the user, this may hurt their impression of the application. As applications can have varying levels of certainty when in use, it may not be wise to always show *certainty* information.

Control. For context-aware applications, this intelligibility type would support users changing parameters in the conceptual model that were originally set by the developers or learned by the system. However, when users adjust such parameters, they may be making poorer choices than the developers or the underlying machine learning algorithm, and may ultimately hurt the application accuracy. Nevertheless, it may be important to allow users to change these settings, but caution them about the danger of doing so.

Design Prescription

We synthesize our findings and discussion to produce an initial attempt at a design prescription on what and how to provide and implement intelligibility types for context-aware applications. We propose a four-step procedure, and use UbiGreen [13] as an example application to illustrate this and for external validity. UbiGreen is a mobile context-aware application that uses a wearable sensor to recognize physical activity relating to sustainability (green) actions. The application tracks the accumulation of green actions and displays a corresponding rewarding wallpaper on the phone throughout the week. The steps are as follows:

I. Map application to moderating circumstances. Determine whether the application will encounter high or low moderator rating values. Table 8 shows the mapping for UbiGreen.

		Intelligibility Type	General	Inappropriate-ness		Criticality		Goal-Supportive		Recommend-ation		Externalities	
				L	H	L	H	L	H	L	H	L	H
Application		Input					1						1
		Output					2				1		
	Model	Why	1	1	1	1	2						
		Why Not			1		1		2				1
		How	1	1	1	1	1			1		1	
		What If					1			1		1	
		What Else					2						
		Visualiza-tion	1	1			1	1			1		
		Certainty	1				2		1				
		Control	1		1		2						
Situation							2						

Table 8: Design prescription of which intelligibility types to implement depending on the circumstances encountered by and functionality of the candidate context-aware application. L=Low, H=High, 1=recommended, 2=highly recommended

Provision Type	Tradeoffs
Always on (always available display)	Obtrusive and space consuming. Only suitable for displays that are persistent (e.g. peripheral or kiosk displays).
Automatic (always show, event-based)	Could be obtrusive for frequent events. May want to deactivate after a while.
Adaptive / Intelligent	The system uses context to determine when to provide explanations. Applications with poor accuracy may provide or omit providing explanations at inappropriate times. This can be used to determine the most crucial time to send privacy-sensitive information.
On Demand$_a$ (always available option)	Least obtrusive, but may not expose user frequently enough to improve their understanding.
On Demand$_e$ (event-based availability)	Allows users to get intelligibility features contextually.

Table 8: Reference of provision types to handle tradeoffs between providing intelligibility and other issues.

Moderator	Lo	Hi
Inappropriateness	✓	✓
Criticality	✓	
Goal-Supportive		✓
Recommendation	✓	
Externalities	✓	

Table 8: Circumstances mapping for UbiGreen.

Issues	Concern
Obtrusiveness	Med
Personal Privacy	Low

Table 9: Issues mapping for UbiGreen.

		Intelligibility Type	Provision	Priority	Details / Comments
Situation			-		
Application		Input	-		
		Output	-		
	Model	Why	On Demand$_e$	Med	To support curiosity or deal with expectation mismatches.
		Why Not	On Demand$_a$	High	Needed to answer why certain activities were not recognized and recorded.
		How	On Demand$_e$	Med	Providing this intelligibility type may be controversial, since this information would allow users to game the system.
		What If	-		Implementing this would also support gaming of the system.
		What Else	-		
		Visualization	(with others)		To augment textual explanations.
		Certainty	Always	Med	Though important to have, it is more appropriate to show certainty at the event level (e.g. detection of various activities), than at the cumulative level (i.e. wall paper display of progress).
		Control	On Demand$_a$	Med	Expose user model and sensor thresholds to allow tweaking.

Table 10: Design prescription for UbiGreen.

II. Referring to Tables 6 and 8, determine which intelligibility types to provide and prioritize them in order of recommendation.

III. Consider issues (such as obtrusiveness and privacy) that would lead to trade-offs with when and how to provide intelligibility. Table 9 lists concerns for UbiGreen, and Table 8 presents ways to provide intelligibility types.

IV. Summarize selections of intelligibility types with justifications. We provide Table 10 as a template for this summary and filled it out with details for UbiGreen.

LIMITATIONS

We acknowledge three limitations of our approach. First, we do not claim our list of moderating circumstances to be comprehensive; there are other explanation types that could be important to context-aware applications (*e.g.*, history, similar examples), and other circumstances (*e.g.*, more vs. less autonomous systems). Second, our participants had to imagine using the various applications through video and text. We used this approach to garner opinion from a larger pool of participants and to investigate a range of multiple context-aware applications. However, this does not mitigate the need to investigate the demand for intelligibility features in deployed systems, where users can get real-time feedback, have internal motivations and goals, and can draw on past experience of using the system. Third, since we drew our participants from the Internet, and a relatively new platform, Mechanical Turk, our population is not a representative sample of the general public (see [25]). Our sample (ethnicity 32% Asian / Pacific Islander, education 37% 4-year college educated, 18% with post-graduate degree; 27% students) is more representative of an internet-savvy population, and should be representative of technology early adopters. This population is appropriate, as we are dealing with the adoption of novel technologies.

CONCLUSIONS AND FUTURE WORK

We have described two experiments about a wide range of scenarios with four context-aware applications to assess user demand for intelligibility and the impact on user satisfaction when those demands are met. From a question asking perspective, we have elicited a set of intelligibility types (*application*, *situation*, *input*, *output*, *model*, *why*, *why not*, *how*, *what if*, *what else*, *certainty*, *control*) users of context-aware systems may be interested in, and several circumstances that may moderate when this demand changes (Inappropriateness, Criticality, Goal-Supportive, Recommendation, Externalities). Our findings suggest that some intelligibility types (*e.g.*, *why*, *certainty*, *control*) should be made available for all context-aware applications, while some are more useful for specific contexts (*e.g.*, *why not* for goal-supportive tasks). We believe that context-aware application developers can take these recommendations on when and how to provide different types of intelligibility features and dramatically improve user satisfaction with, and acceptance of, their context-aware applications.

We have found that even if certain intelligibility types are helpful to improve users' understanding, they may not necessarily want or seek them, and be more satisfied. We would like to investigate the interactions involved in increasing understanding and satisfying information demands to provide a more cohesive framework for intelligibility. We would like to test our findings in a field study with a deployed intelligible context-aware application. A fully-functional (or Wizard-of-Oz) intelligible application will be able to support wider varieties of explanations for each intelligibility type (such as *why not*), and provide more relevant responses to the user. We have also recommended various delivery mechanisms (on demand, adaptive, and automatic) for providing intelligibility features. We would like to investigate the trade-off between effectiveness and obtrusiveness of each provision type.

ACKNOWLEDGEMENTS

This work was funded in part by the National Science Foundation under grant 0746428, and the Agency for Science Technology And Research, Singapore.

REFERENCES

1. Abowd, G.D., Atkeson, C.G., Hong, J., Long, S., Kooper, R. & Pinkerton, M. (1997). Cyberguide: a mobile context-aware tour guide. *Wireless Networks* 3(5), 421-433.
2. Abowd, G.D., Mynatt, E.D. & Rodden, T. (2002). The Human Experience, *IEEE Pervasive Computing* 1(1), 48-57.
3. Antifakos, S. Kern, N., Schiele, B. & Schwaninger, A. (2005). Towards improving trust in context-aware systems by displaying system confidence. *Proc. MobileHCI 2005*, 9-14.
4. Avrahami, D. & Hudson, S.E. (2006). Responsiveness in Instant Messaging: Predictive Models Supporting Inter-Personal Communication. *Proc. CHI'06*, 731-740.
5. Barkhuus, L. & Dey, A.K. (2003). Is context-aware computing taking control away from the user? Three levels of interactivity examined. *Proc. Ubicomp 2003*, 149–156.
6. Bellotti, V. & Edwards, W.K. (2001). Intelligibility and Accountability: Human Considerations in Context-Aware Systems, *Human-Computer Interaction*, 16(2-4): 193-212.
7. Cheverst, K., Byun, H.E., Fitton, D., Sas, C., Kray, C. & Villar, N. (2005). Exploring issues of user model transparency and proactive behavior in an office environment control system. *User Modeling and User-Adapted Interaction* 15(3-4), 235-273.
8. Dey, A.K. & Abowd, G.D. (2000). CybreMinder: A Context-Aware System for Supporting Reminders. *Proc. Handheld and Ubiquitous Computing*, 172-186.
9. Dey, A.K., Abowd, G.D. & Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2–4): 97–166.
10. Dey, A.K., Sohn, T., Streng, S. & Kodama, J (2006). iCAP: Interactive Prototyping of Context-Aware Applications. *Proc. Pervasive 2006*, 254-271.
11. Dzindolet, M., Peterson, S., Pomranky, S. Pierce, L. & Beck, H. (2003). The role of trust in automation reliance, *International Journal of Human-Computer Studies*, 58(6): 697-718.
12. Fogarty, J., Hudson, S.E., Atkeson, C.G., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J.C. & Yang, J. (2005). Predicting human interruptibility with sensors. *ACM Transactions on Computer-Human Interaction* 12(1), 119-146.
13. Froehlich, J., Dillahunt, T., Klasnja, P., Mankoff, J., Consolvo, S., Harrison, B., and Landay, J. A. (2009). UbiGreen: investigating a mobile tool for tracking and supporting green transportation habits. *Proc.CHI 2009*, 1043-1052.
14. Glass, A., McGuinness, D. & Wolverton, M. (2008). Toward establishing trust in adaptive agents. Proc.IUI'08. 227-236.
15. Graesser, A.C. & McMahen, C.L. (1993). Anomalous information triggers questions when adults solve quantitative problems and comprehend stories. *Journal of Educational Psychology*, 85, 136-151.
16. Gregor, S. & Benbasat, I. (1999). Explanations From Intelligent Systems: Theoretical Foundations and Implications for Practice. *MIS Quarterly* 23(4): 497–530.
17. Herlocker, J., Konstan, J. & Riedl, J. (2000). Explaining collaborative filtering recommendations. *Proc. CSCW 2000*, 241-250.
18. Ko, A. J. & Myers, B. A. (2009). Finding causes of program output with the Java Whyline. *Proc. CHI '09*, 1569-1578.
19. Kuleza T., Wong, W.K., Stumpf, S., Perona, S., White, R., Burnett, M.M., Oberst, I. & Ko. A.J. (2009). Fixing the Program My Computer Learned: Barriers for End Users, Challenges for the Machine. *Proc. IUI 2009*, 187-196.
20. Lim, B. Y., Dey, A. K. & Avrahami, D. (2009). Why and why not explanations improve the intelligibility of context-aware intelligent systems. *Proc. CHI 2009*, 2119-2128.
21. McGuinness, D. L., Glass, A., Wolverton, M. & Pinheiro da Silva, P. (2007). A Categorization of Explanation Questions for Task Processing Systems. *AAAI Workshop on Explanation-Aware Computing (ExaCt-07)*.
22. 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.
23. Mynatt, E. D., Rowan, J., Craighill, S. & Jacobs, A. (2001). Digital family portraits: supporting peace of mind for extended family members. *Proc. CHI '01*, 333-340.
24. Orlikowski, W. J. (2000). Using Technology and Constituting Structures: A Practice Lens for Studying Technology in Organizations. *Organization Science* 11(4), 404-428.
25. Schonlau, M., Soest, A. V., Kapteyn, A., & Couper, M. P. (2006). Selection bias in web surveys and the use of propensity scores. *Sociological Methods & Research*, 37(3): 291-318.
26. Strauss, A.L. & Corbin, J.M. (1990). Basics of Qualitative Research: Grounded theory procedures and techniques. Newbury Park, CA: Sage Publications.
27. Van der Meik, H. (1987). Assumptions of information-seeking questions. *Questioning Exchange* 1, 111-118.