Field Evaluation of an Intelligible Context-Aware Application

Brian Y. Lim, Anind K. Dey

March 8, 2011
CMU-HCII-12-102

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, predictions, instant messaging, context-awareness, human-computer interaction.

ABSTRACT

Context-aware applications can facilitate people as they carry out their daily tasks. These applications can use a suite of sensors to detect what is happening in the environment and with the user. They can then infer the user intention. This way, they try to understand the contexts of the situation, and consequently act to provide services. For example, a smart phone can recognize that you are in a conversation, and suppress any incoming messages during this period. To minimize obtrusiveness and allow users to focus primarily on their tasks, context-aware applications perform sensing implicitly without explicitly informing users. Furthermore, to better understand the contexts of users in their physical and social environments, context-aware applications are using increasingly complex mechanisms to infer these contexts (*e.g.*, by using machine learning algorithms). This implicit sensing and complex inference can remain invisible when the applications work well and as expected, but become a mystery when the applications behave inappropriately or unexpectedly. In such cases, the lack of understanding of these applications can lead users to mistrust, misuse it, or abandon them altogether. To counter this, context-aware applications should be *intelligible*, capable of generating explanations of their behavior.

INTRODUCTION

Context-aware applications can be difficult for users to make sense of because they are based on implicit sensing and inferences and often take invisible actions. Providing explanations is a way to expose the inner workings such that these applications become *intelligible* [3] to users, helping users to both develop more accurate mental models of these systems and increase their trust in these applications. However, many context-aware applications have to operate in a wide variety of situations for a wide range of users. This may cause them to make incorrect inferences and take incorrect actions. We are interested in knowing whether providing explanations affect how users trust the applications; whether explanations help or hurt the user experience in the face of inaccurate context-aware inferences; how much users would use explanations; and whether they prefer some explanation types over others.

We extend research into intelligibility by investigating these issues through the use of an intelligible context-aware application in a longitudinal field study. Past research on intelligibility in context-aware applications has mainly consisted of lab studies (*e.g.*, [6, 7]), and this work seeks to validate their results in a field study for increased external validity. We studied the interaction between inference accuracy and explanation provision in a three-week field study with 100 participants who used an intelligible, novel, context-aware instant messaging plugin application that predicts one's response time to a buddy's message. The application is able to provide explanations of how it derived its prediction, including certainty [1] and answers about why a particular application or user feature is relevant to the prediction (why explanations) [6]. Users either received explanations every time there was a prediction (Always On), when they asked for them (On Demand), or never (None). The application was more accurate when predicting shorter response times than when predicting longer ones. We found that users trusted the predictions more when they received explanations. We also found that they used explanations more for inaccurate predictions, although the explanations increased trust more when presented with accurate predictions. We recommend that system builders include explanation when their system can perform inferences accurately, and should otherwise strive to increase the accuracy of their systems before deploying intelligibility support.

In this paper we contribute an increased understanding of the value of explanations for improving trust in context-aware explanations, particularly in the face of application inaccuracies. In the remainder of the paper, we describe the intelligible application prototype we deployed for the study, the user study to investigate the interaction between accuracy and intelligibility, our findings, and some recommendations on how to target the use of explanations.

INTELLIGIBLE PROTOTYPE: IM AUTOSTATUS PLUGIN

Interruptibility [2, 4] and awareness are of much interest in the research community, particularly with people attempting to juggle many tasks at the same time to optimize their efficiency. In particular, when engaged in instant messaging, one may wish to know when a buddy will respond so that one may spend time on another task if the buddy is not going to respond for a while.

We developed the IM Autostatus application, a plugin to the AOL Instant Messenger (AIM) Windows client that a *receiver* buddy installs. When a *sender* buddy sends a message to a receiver, the plugin responds with a prediction of when the receiver will respond. The application has three class values to predict: whether the user would respond (i) within one minute ($t \leq 1$ min), (ii) between one and five minutes ($1 \text{ min} < t \leq 5$ min), and (iii) after five minutes ($t > 5$ min). Predictions are not provided when $t \leq 1$ min, so as not to be too obtrusive, but are shown at other times. The application is based on the statistical model of responsiveness developed by Avrahami and Hudson [2], and uses the Subtle toolkit [4] to sense several events on the receiver's computer (see **Error! Reference source not found.**).

IM Events	Desktop Events
Message sent / received	Key press
Message window open / close	Mouse button click
Status changes (online, away, *etc.*)	Window created, minimized, in focus, *etc.*

Table 1. Several IM and desktop events detected by the IM Autostatus application. Adapted from [1].

The statistical model was simplified to use a smaller feature space than [2], in order to function in real time, without significantly compromising prediction performance. We pre-trained the model on the dataset provided from [2] rather than involve our participants in a lengthy training process (3-4 weeks long). The IM Autostatus plugin application is intelligible and able to explain how it makes its predictions. Using an underlying decision tree statistical model, the application can generate *why* [6] and *certainty* [1] explanations at run time. Why explanations describe the logical flow through the decision tree, and explain why the application made its prediction at the time of inquiry. They are generated by walking through the tree. The full decision tree model trained for the application had a depth of 26 and 5315 leaves. A full why explanation that walks the full depth of the tree would provide an excessively long explanation. We shortened the explanation by revealing the 5 most important, human understandable features (and their values) that led to the prediction, similar to that of [8]. Certainty explanations take the confidence values the decision tree model has at the leaf of its current decision, presented as a probability percentage of how likely the outcome is.

We also provided three ways that explanations would be delivered: Always On (AO), On Demand (OD) and None (N). OD explanations are supported by a prompt grammar that can be used to interact with the application (see Figure 1 for an example) after receiving a prediction message.

Figure 1. Screenshots of the plugin's responsiveness prediction, and receiving why, more detailed, and certainty explanations.

Why explanations are requested by typing `im-why` (lists the 5 most relevant features) with `im-1`, `im-2`, *etc.*, for the value of the respective features, and certainty explanations by `im-%`. AO `im-why` and `im-%` explanations are provided whenever a prediction message is shown and users cannot control whether to see or suppress them. `im-#` explanations are optional. The None version of the application does not provide explanations.

HYPOTHESES

We believe that explanations are a useful way to indicate to users how a context-aware application works and to support users in fostering trust in these applications. We have the following hypotheses:

H1a: None < {On Demand & Always On}. Users would trust the application more when receiving explanations.

H1b: On Demand < Always On. Users would have greater trust when exposed to more explanations

However, we also hypothesize that the accuracy of the functionality (prediction) would influence the impact of explanations on trust:

H2: Explanations will not be as useful for less accurate functionality, because users may not be convinced by the argument of an already poorly perceived application.

METHOD

To test our hypotheses, we deployed the IM Autostatus plugin in a field study for up to 3 weeks (average of just over 1 week, due to high attrition rate) per participant. Participants were recruited in groups of at least 3 buddies, where one buddy is the *receiver* who installs the application and the others are *senders* who view the application predictions and explanations. Groups were distributed across provision types between-subjects.

Procedure and Measures

At the beginning of the study, the receiver installs the plugin. It is instrumented to only send IM Autostatus messages to buddies who are in the study (senders), and only collect log data from the receiver and senders. Log data includes feature data from the receiver's computer for every prediction made (*i.e.*, when messages are received from sender buddies), and records of the actual response times (*ground truth* measure for prediction accuracy). Up to four times a day, sender buddies are asked how much they agree with a prediction just made by the plugin. We use the ***agreement*** measure (5-pt Likert scale: strongly disagree to strongly agree) as a measure of user trust in the application. We also conducted follow-up interviews with a subset of participants to understand their use of the explanations.

Category	Type	Description	κ	#
No idea / guess	*guess, wrong*	No reason given, or stated guessed	.93	.200
Content analysis	*wrong*	*E.g.*, vocabulary, text analysis	1.00	.104
IM status / idle	*common sense*	Status or time being online, away, idle	.94	.144
IM history	*common sense*	Chat history, timings of previous IMs	.98	.336
IM average timings	*common sense*	Average response times	1.00	.168
Keyboard typing	*application*	That keyboard activity is logged	1.00	.096
Demographic	*application*	Age, gender (plugin uses these)	1.00	.088
Window focus	*application*	Whether / which window has focus	.93	.056
# of Windows	*application*	Number of windows opened	.97	.200
Has URL	*rare*	Whether IM message has a URL	1.00	.032

Table 2. Coding scheme for reasons in participants' mental models about the IM Autostatus plugin. Inter-rater reliability (κ) between two coders is shown, and the average number of respective reasons in each mental models (#). Type represents which higher level codes the category is assigned to (see text for explanations).

We ask participants to fill out a weekly survey in which they describe how they think the plugin works, *i.e.*, their mental models. Two raters coded and counted the reasons using the coding scheme shown in Table 2.

The codes were condensed to counts of the following: **reasons** that were not guesses; **correct reasons**; **wrong reasons**; **no idea** / guess; **common sense reasons**, given a reasonable understanding of instant messaging; **application-specific reasons** about the plugin that would be difficult to guess; and **rare** application-specific reasons (rare).

RESULTS

We recruited 100 participants from online sources (a local experiment scheduling site, Craigslist, and Mechanical Turk), in 31 groups with 31 receivers and 69 senders (ages 18 to 53, Median=23.5; 45% females; IM usage 4 to 7 days/week, Median=7 days, 2 to 12 hrs/day, Median=4.5 hrs). Participants were paid $10 per week of participation.

We filtered the agreement responses to keep those that were completed before the receiver buddies responded to avoid possible cheating. We further filtered the remaining responses for those that took too long (>5 min) to be completed such that the participant may have forgotten the original context of the questions. This left us with 155 valid responses from the original 533. Some participants using the OD version of the application never asked for explanations, so we treated their data as if in the N condition. An analysis of the log data for all participants revealed that the accuracy of predictions were 82.2% (for $t \leq 1$ min), 32.0% ($1 < t \leq 5$), 11.9% ($t > 5$), p<001. We combine cases where the application predicted $t > 1$ min as "Low accuracy" and $t \leq 1$ min as "High accuracy."

Agreement with Application Prediction

We dichotomized (0 or 1) the agreement 5-pt Likert scale measure to a binary variable: "not agree" (1-3) and "agree" (4-5) to minimize the variance of how individuals used the Likert scale. We conducted a two-way multi-level ANOVA with Provision Type and Accuracy as main effects, and Provision Type × Accuracy as an interaction effect. Participants agreed with the plugin more for predictions of $t \leq 1$ min than $t > 1$ min (M=.86 *vs.* M=.57, p<.001). A one-tailed contrast of N < {OD & AO} (*H1a*) was significant overall (p=.025), and especially for High accuracy (one-tailed, p=.042). A one-tailed contrast of OD < AO was marginally significant overall (p=.1) and for High accuracy (one-tailed, p=.061) (*H1b*). To test H2, we performed two contrasts comparing the difference in the improvement of agreement for N *vs.* {OD & AO} between high and low accuracy. We found that the difference is smaller for low accuracy than high accuracy (p=.03; see Figure 2).

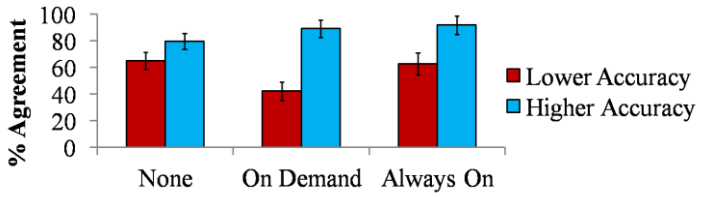

Figure 2. Agreement with plugin across provision type and accuracy. Participants have a higher increase in agreement when receiving explanations for higher accuracy.

On Demand Explanation Usage

We analyzed participant logs to determine how they ask for explanations. Participants receiving explanations AO asked more detail questions (*i.e.*, more usage of `im-#`) than those using OD explanations (M=6.5%, n=148 *vs.* M=.1%, n=29, p=.05). For participants with OD explanations, they asked for more why explanations than certainty ones (M=1.6%, n=152 *vs.* M=.4%, n=93, p<.001). They also asked for more explanations for lower accuracy predictions than the higher accuracy ones (M=3.9%, n=94 vs. .7%, n=75, p=.04). Analyzing usage over time (≤5 *vs.* >5 days), we found that more explanations were asked earlier than later (M=4.0% *vs.* M=.2%, p<.001), and usage dropped more sharply for higher accuracy than lower

accuracy predictions (p=<.001). A 2-way ANOVA of explanation type (why *vs*. certainty) and prediction class (3 levels of time responsiveness) found significant differences for prediction class as a main effect and particularly for the interaction effect of explanation type × prediction (p<.001). The longer the predicted response time, the more times participants asked for why explanations, while there were no differences in the number of times participants asked for certainty.

Mental Models

125 mental models were coded by two raters (see reliabilities in Table 2). On average, each description of how the plugin works contains 1.69 reasons, 1.55 correct ones, and .13 wrong ones. Participants were modeled as a random effect and nested in groups in an ANOVA analysis with Provision type as a main effect. Participants receiving explanations (OD or AO) had more correct reasons than those not receiving explanations (N) (p=.02). The number of incorrect reasons was not moderated by provision type (p=n.s.). Mental models from participants receiving OD or AO explanations contained more application-specific reasons (p=.001). Participants receiving AO explanations reported more rare application-specific reasons compared to other provision types (p=.02).

DISCUSSION AND RECOMMENDATIONS

The results substantiate some of our hypotheses, and reveal some new insights. Hypothesis 1a (H1a) was supported: explanations helped to improve agreement with the novel context-aware application. Indeed, one participant was fascinated with what the plugin could do and how it could do it, stating that the explanations give the predictions a "scientific" basis. For H1b, there was no difference in agreement across explanation provision types, but there was a marginally significant difference between OD and AO for higher accuracy predictions (Figure 2). This limited benefit of AO explanations may be due to participants ignoring the messages after a while. H2 was supported: explanations improved trust more for high accuracy predictions than low accuracy ones. One participant stated that there was no point reading the explanations if the predictions were already wrong.

Targeted Intelligibility

In agreement with [7], we found that explanations would be more useful to specific users (with a greater social distance [8]) and situations (higher accuracy), and that certain explanations are more useful than others for this application (*why* more popular than *certainty*). One participant pointed out that she did not find explanations particularly useful since she already knows a lot about her buddy. A receiver indicated that she was not interested in seeing information about herself since she was already self-aware. For such a non-critical, social, context-aware application (like the IM Autostatus plugin), explanations are more useful if about *someone else* whom the user is *less familiar* with.

Participants used explanations more and had greater trust when the accuracy of the plugin was higher. This suggests that application designers should include intelligibility *and* continue to strive to make their applications more accurate.

Participants indicated that they either used explanations out of *curiosity* or to learn how the plugin worked, and to conduct *fault finding* when they felt the plugin was wrong (more usage of *why* explanations for lower accuracy predictions). Participants did not use *certainty* explanations more when investigating inaccurate predictions. This validates lab results from [6] and [7] that why explanations are the most important explanation type to provide.

Efficacy of Intelligibility Provision Types

Analysis of the mental models of participants indicates that explanations do help participants understand the technical capabilities of a context-aware application and that they can remember application-specific and

even rare reasons. Explanations needed to be Always On for users to encounter rare reasons. However, many participants were not happy about the obtrusiveness of the long explanations, with one group stating this as their main reason for dropping out of our study. Always On explanations interfered with chatting and reviewing chat histories.

CONCLUSION AND FUTURE WORK

We deployed an intelligible context-aware IM Autostatus plugin that predicts when one's buddy would respond to a message and is able to explain its inferences. We ran a user study to investigate the effects of intelligibility (None provided or provided On Demand or Always On) on higher and lower accuracy predictions. We found that explanations helped to reinforce trust (agreement with the predictions) more for more accurate predictions than for less accurate predictions. Participants had higher trust and learning outcomes when receiving explanations Always On, but did not like the obtrusiveness of that provision type. We recommend that explanations be provided when accuracy is high, and that developers strive to increase the accuracy of their systems before deploying intelligibility support.

In this study, we investigated Always On and On Demand provisioning of explanations. Others have proposed an Intelligent provision type, in which the system decides when to provide an explanation [5]. We will investigate this in the future, as context-aware applications can use context to intelligently determine when best to provide explanations. Perhaps intelligently providing explanations can provide an automatic balance for information maximization and obtrusiveness minimization. However, there is also a danger that applications may provide explanations at the wrong time.

ACKNOWLEDGEMENTS

We thank Andreas Möller for his work in implementing the IM Autostatus plugin.

REFERENCES

1. Antifakos, S. Kern, N., Schiele, B., and Schwaninger, A. (2005). Towards improving trust in context-aware systems by displaying system confidence. *Proc. MobileHCI 2005*, 9-14.
2. Avrahami, D. and Hudson, S. E. (2006). Responsiveness in instant messaging: predictive models supporting inter-personal communication. *Proc. CHI '06*, 731-740.
3. Bellotti, V. and Edwards, W.K. (2001). Intelligibility and Accountability: Human Considerations in Context-Aware Systems, *Human-Computer Interaction*, 16(2-4): 193-212.
4. Fogarty, J. and Hudson, S. E. 2007. Toolkit support for developing and deploying sensor-based statistical models of human situations. *Proc. CHI '07*, 135-144.
5. Gregor, S. and Benbasat, I. (1999). Explanations From Intelligent Systems: Theoretical Foundations and Implications for Practice. *MIS Quarterly* 23(4): 497–530.
6. Lim, B. Y., Dey, A. K., and Avrahami, D. (2009). *Why and why not* explanations improve the intelligibility of context-aware intelligent systems. *Proc. CHI '09*, 2119-2128.
7. Lim, B. Y. and Dey, A. K. (2009). Assessing Demand for Intelligibility in Context-Aware Applications. To appear in *Proc. Ubicomp 2009*.
8. Olson, G. M. and Olson, J. S. (2000). Distance matters. *Human-Computer Interaction*. 15, 2, 139-178.
9. Tullio, J., Dey, A. K., Chalecki, J., and Fogarty, J. 2007. How it works: a field study of non-technical users interacting with an intelligent system. *Proc. CHI '07*, 31-40.