User Modeling and User-Adapted Interaction (2005) 15:235–273 © Springer 2005
DOI 10.1007/s11257-005-1269-8

Exploring Issues of User Model Transparency and Proactive Behaviour in an Office Environment Control System

KEITH CHEVERST, HEE EON BYUN, DAN FITTON, CORINA SAS, CHRIS KRAY and NICOLAS VILLAR
Department of Computing, Lancaster University, Lancaster, LA1 4YR, UK.
e-mail: kc@comp.lancs.ac.uk

(Received: 4 April 2004; accepted in revised form: 5 June 2005)

Abstract. It is important that systems that exhibit proactive behaviour do so in a way that does not surprise or frustrate the user. Consequently, it is desirable for such systems to be both personalised and designed in such a way as to enable the user to scrutinise her user model (part of which should hold the rules describing the behaviour of the system). This article describes on-going work to investigate the design of a prototype system that can learn a given user's behaviour in an office environment in order to use the inferred rules to populate a user model and support appropriate proactive behaviour (e.g. turning on the user's fan under appropriate conditions). We explore the tension between user control and proactive services and consider issues related to the design of appropriate transparency with a view to supporting user comprehensibility of system behaviour. To this end, our system enables the user to scrutinise and possibly over-ride the 'IF-THEN' rules held in her user model. The system infers these rules from the context history (effectively a data set generated using a variety of sensors) associated with the user by using a fuzzy-decision-tree-based algorithm that can provide a confidence level for each rule in the user model. The evolution of the system has been guided by feedback from a number of real-life users in a university department. A questionnaire study has yielded supplementary results concerning the extent to which the approach taken meets users' expectations and requirements.

Key words. context history, intelligent environment, inference, machine learning, proactive behaviour, prototype deployment, scrutability

1. Introduction and Motivation

Over the past decade, one of the predominant trends in computing has been *ubiquitous computing*, the term having first been proposed by Weiser in the early 1990s. His vision included the notion of increasing the productivity or welfare of a user situated in a *computer-everywhere* environment by supporting human assistance in an intimate way (Weiser, 1991). One research domain that requires the computer-everywhere model of ubiquitous computing is that of the *intelligent environment* (Coen, 1998). In this domain, a wide range of physical devices (e.g., lights, audio/video equipment, heaters, air conditioning equipment, windows, curtains, etc.) can be controlled automatically on the basis of the context of a house or office and the preferences of the inhabitants (e.g., preferred temperature, preferred light level, energy consumption policies, etc.). One promising technique for achieving such intelligent environments is *context-aware computing*. Context-aware systems have been defined by Dey and Abowd (2000) as:

> "systems [that] adapt according to the location of the user, the collection of nearby people, hosts, and accessible devices, as well as to changes to such things over time".

However, as is described by Cheverst et al. (2001), when designing context-aware systems designers need to be conscious of a number of potential pitfalls in order to ensure that the context-aware system maintains appropriate levels of predictability and provides sufficient *transparency* to enable users to trust the behaviour of the system, to understand what adaptation strategies are in place, and to over-ride these adaptation strategies in those circumstances where such user control is warranted. As is noted by Jameson et al. (2004), the term *transparency* is an overloaded term in the literature on human-computer interaction. For the remainder of this article, we will use the more specific term *comprehensibility* to suggest that the user:

> "... can look through the outer covering (e.g. a glass box) to examine the inner workings of the device."

Concerning this need for comprehensibility, Abowd and Mynatt (2000) make similar comments when discussing key challenges in the ubicomp domain:

> "One fear of users is the lack of knowledge of what some computing system is doing, or that something is being done 'behind their backs'".

Unfortunately, one implication of signalling to the user that the system is 'doing something behind their back' is the potential for inappropriate interruption of the user's current task or activity. The negative impact of interruptions on task performance, particularly due to the shift of attention and memory load, has been widely documented in various studies. Despite the impact of interruptions on task performance, it has been hypothesised that a well-designed user interface may compensate for such negative effects and exploit the potential of multitasking which, with appropriate support, people are capable of. For an excellent review of this work see McFarlane and Latorella (2002).

Scrutability refers to the ability of a user to interrogate her user model in order to understand the system's behaviour. In relation to ubicomp environments, Kay et al. (2003) describe how:

> "...when the user wants to know why systems are performing as they are or what the user model believes about them, they should be able to scrutinise the model and the associated personalisation processes."

Relating scrutability to the issue of control, Kay et al. (2003) also state that:

"...one of the important requirements of ubiquitous computing, [is] that of ensuring user control over the model"

and

"We see scrutability as a foundation for user control over personalization".

In this article, we describe our formative work on exploring the comprehensibility, scrutability and control issues associated with the on-going development and evaluation of an *Intelligent Office System*, hereafter referred to as IOS. The system can learn a given user's behaviour in an office environment in order to use inferred rules to populate his or her user model and support appropriate *proactive* behaviour, such as turning on the user's office fan under appropriate contextual conditions (Byun and Cheverst, 2004). When we use the term *proactive*, we agree with the understanding of the term presented by Salovaara and Oulasvirta (2004) whereby:

"Proactive systems adhere to two premises: 1) working on behalf of, or pro, the user, and 2) acting on their own initiative."

Our investigation is interested in uncovering and exploring:

- The possible techniques and surrounding issues for making the behaviour of the system comprehensible to the user and the impact this can have on the selection of underlying machine learning approach.
- Issues surrounding how users can be given appropriate levels of *control* while still avoiding undue levels of interruption.

The IOS infers its rules from the context history (effectively a data set generated using a variety of sensors) associated with the user. Part of our original motivation was to determine whether it would be possible for context history to be used to infer the nuances of a given user's behaviour. For example, with regard to the temperature inside an office, there may be numerous ways in which a user might attempt to cool the temperature. She might open the door (if it is cooler outside), close the blind (if strong sunlight is shining into the office), turn on the fan (if she is not engaged in an activity where the noise of the fan would be distracting), open a window (if it is not too noisy outside) etc. Our hypothesis was that by having an appropriate set of sensors recording context such as office temperature, level of light, etc., it would be possible for a system to learn the user's favoured approach for controlling the office temperature and then (to an extent acceptable to the user) automate the process, e.g. by turning on the fan.

The system that we have developed learns from a context history using either a standard or a fuzzy decision-tree-based algorithm; the latter is capable of providing the user with an idea of the confidence level associated with the inferred rules. Adopting this decision tree approach enables users to inquire into the behaviour of the system by being allowed to scrutinise and possibly override the 'IF-THEN' type rules held in her user model. Currently, a number of users in a university department are running the system (the software for which is publicly available) in order to explore the feasibility of the approach and user acceptance. We have also carried out a questionnaire-based survey (completed by 30 participants) in order to obtain further insights into the levels of comprehensibility and control sought by its users.

The remainder of this article is structured as follows. In the following section, we discuss related work in this particular application domain. Next, in Section 3, we discuss in detail our approach both in terms of research methodology and technical decisions (including our use of decision tree learning and use of fuzzy logic). The IOS is of course composed of both software and hardware components (not forgetting the user). In Section 4, we describe the hardware used by the system to sense the user's environment and actuate supported devices. Then Section 5 describes, by way of a walkthrough, the user interface to the IOS. In Section 6, we describe an analysis of the findings to date, including lessons learned and the results of the questionnaire-based user survey. The penultimate section on future work is followed by a section presenting a summary and concluding remarks.

2. Related Work

This section initially outlines some representative proactive systems developed for home environments, which particularly address the issues of comprehensibility and control. We continue by describing research projects which specifically focus on the relationship between user control and service automation. Finally, we discuss issues related to interruptibility (arsing in the context of proactive behaviour).

2.1. USER CONTROL IN AUTOMATED/CONTEXT-AWARE SYSTEMS

The aim of the Adaptive House project (Mozer and Miller, 1998) at the University of Colorado, Boulder, was to make a home that can adjust its functions to the schedules and lifestyles of the inhabitants. A house in Boulder, Colorado was fitted with more than 75 sensors capable of sensing the physical environment in the house. Data were gathered, for example, on room temperature, light level, sound level, and door and window positions. The developers combined two machine learning algorithms in order to realise adaptive behaviour. First, an artificial neural network attempts to predict the likely location of an inhabitant in the next two seconds. Second, a reinforcement learning algorithm determines an action (e.g., turning lights on/off and setting their intensities) that has the minimum expected *discounted cost* with respect to the prediction of the inhabitants' location. Unlike our approach, the behaviour of the Adaptive House control system cannot be scrutinised, nor can the user explicitly influence it.

The EasyLiving Project (Brumitt et al., 2000) was concerned with the development of an architecture and suitable technologies to enable typical PC-focused activities to move from a fixed desktop into the environment as a whole. The project focused on the technologies of middleware, geometric world modelling, perception, and service description. Furthermore, the project team implemented several applications to operate within an intelligent space where a user model of each person is maintained, storing, for example, the user's physical appearance, authentication instructions, and also personal preferences. One application, 'Media Control', was capable of proactively playing various media types (for example, a CD, MP3, DVD or Videotape) based on the user's preferences and current location.

Intille (2002) designed the Changing Places/House_n, a full-scale single-family home with an integrated and ubiquitous sensor architecture, as a model home for the future. The house was outfitted with a computer-controlled heating, ventilating, air-conditioning (HVAC) system, but it was not used to automate fully control of the condition of the environment. Instead, the system provides subtle reminders to the inhabitants to perform a certain action. For example, the system will not open the window by itself but will turn on a tiny light on the window in order to inform the user that it is the time to open the window. In this scenario, Intille raised two important issues that are closely related to our research (and which we discuss in more detail in Sections 6.2.3.1 and 7.4):

- There is an inherent tension between the amount of control the user has over a system and the benefits she may enjoy from proactive services.
- The suggestion to be presented to a user should not divert the user's attention from their current task.

With regard to these issues, Intille (2002) argues that the complexity of algorithms for making a decision (e.g. opening the window) may prevent the user from understanding the behaviour of the system and could therefore be perceived as "unexplainable intelligence".

The three systems outlined above mark distinct areas within a design space along the two dimensions of control and scrutability. Control may vary from complete system control to full user control, whereas scrutability varies from low to high (see Figure 1).

The IOS, which is the focus of this article, offers high scrutability and allows the user to exert a high level of control.

The following systems represent cases of supporting proactive behaviour based on some model of the user or inhabitants. Mitchell et al. (1994) developed the Calendar APprentice (CAP) application to assist a user with calendar scheduling based on the user's scheduling preferences. In particular, through the design of the CAP, the researchers explored the potential of machine learning methods for the implementation of personal software assistants. Each night, the CAP system runs a decision tree learning algorithm on a chunk of the user's most recent calendar information, in order to refine the set of rules that will be used to provide advice on the following day. The work described in this article extends the approach

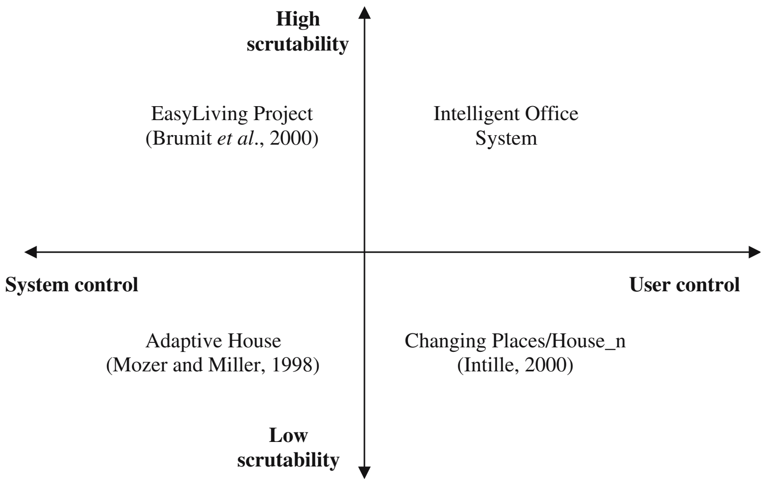

High scrutability
EasyLiving Project (Brumit *et al*., 2000)
Intelligent Office System
System control
User control
Adaptive House (Mozer and Miller, 1998)
Changing Places/House_n (Intille, 2000)
Low scrutability

Figure 1. Two-dimensional design space spanning control and scrutability dimensions.

presented by Mitchell et al. by using fuzzy decision tree learning to provide users with both user-confirmed and fully automated proactive behaviour for multiple services.

Barkhuus and Dey (2003) report on findings from a study in the context of a mobile scenario, where they investigated the relationship between user control and service automation. In their particular setting, mobile phone users were willing to give up control for a number of (future) services such as tracking the location of friends or recommendation of nearby restaurants at lunch time. Although their study relied on the participants to imagine their usage patterns if such a service was available, one of their main findings was that users were willing to give up control if the benefits (i.e. the convenience or added value) of doing so was high.

However, there may be further, less obvious, drawbacks to giving up control. For example, Intille and Larson (2003) cite Rodin and Langer (1977) to point out that "lack of control over aspects of life has been shown to diminish health". This is the main reason why, in the Changing Places/House_n project, they decided to provide subtle clues to the user to perform an action (e.g., a glowing light switch to suggest turning on the light) instead of a fully automated system.

In this special issue, Carmichael et al. describe their accretion-resolution user modeling representation, which enables a user to scrutinise her user model, the processes that determine its content, and the way that it is used in the ubiquitous computing environment. Their approach is also capable of representing other entities (e.g., sensors), and resolving (by assigning reliability metrics) the fact that certain sensors (e.g., location sensors) may provide conflicting evidence regarding the user's context at a given time.

2.2. INTERUPTABILITY

A further issue in the context of proactive behaviour is the kind and impact of the *interruption* that occurs when a system performs a proactive action or raises the user's awareness of a planned proactive action.

There is a negative relationship between interruption frequency and performance on complex tasks (Speier et al., 1997). In addition, switching between tasks comes with a time cost that increases with task complexity and unfamiliarity (Rubinstein et al., 2001). McFarlane and Latorella (2002) suggest that timing of an interruption must be context-sensitive.

There appear to be individual differences in the ability of people to accommodate interruptions, and this fact must be accounted for in an intelligent system (McFarlane and Latorella, 2002). Therefore, the adaptivity of such a system should be tuned to a user's profile in terms of her behaviour patterns and her ways of handling interruptions. The individual differences in terms of sensitivity to different modalities appear to have greater impact on the disruption caused by interruptions than the interruption modalities themselves, i.e. heat, smell, sound, vibration, light (Arroyo et al., 2002).

McFarlane and Latorella (2002) suggest various methods for responding to interruptions and discuss the following four basic solutions for coordinating interruptions: (i) as soon as the system identifies the need for proactive behaviour (*immediate interruptions*); (ii) at a moment specified by the user after previous negotiation with the system (*negotiation*); (iii) at a moment decided on by the system as appropriate for interruption (*mediated*); or (iv) all the interruptions can occur at once at a previously arranged moment (*scheduled*). The findings in McFarlane and Latorella (2002) obtained through studying the effects of interruption on tasks carried out on a desktop computer suggest that negotiation is the best solution in terms of task performance for almost all situations, but for those where the timing for handling the interruption is critical, the best solution is the immediate interruption.

An excellent analysis of the critical factors surrounding the effective integration of automated services with direct manipulation interfaces is provided by (Horvitz, 1999). It includes the challenge of decision making under uncertainty (e.g., about the user's current goals), the need to consider the status of a user's attention in the timing of services, and "employing dialog to resolve key uncertainties" while "considering the costs of potentially bothering a user needlessly".

3. Approach

3.1. OVERVIEW OF THE TECHNICAL APPROACH

In order to ascertain the feasibility of supporting modelling-based proactive adaptations in a user's office environment, our current work has involved the design and implementation of a system to:

- Utilise context history in order to learn the patterns of the user's behaviour in her physical office environment.
- Support modelling-based proactive adaptations (opening/closing the window, turning on/off the fan, etc.) based on both the patterns learned (represented as a set of rules) and the state of the physical office environment (realised through a set of sensors).

The contexts considered in the experiment are: temperature, humidity, noise level, light level, the status of the window, the status of the fan, and the location of a user. Our system collects and accumulates the contexts as a context history. An early version of the system required the user to notify the system explicitly of instances when she turned on/off the fan or opened/closed the window in her office. Unsurprisingly, initial user trials showed that this task was simply too arduous for the user (and hardly in the spirit of ubicomp). Therefore, we have redesigned the system to collect context automatically from the user's physical environment if she has the appropriate sensors available.

Once a context history has been gathered, a set of human comprehensible rules is induced which represent the user's preferences for the activation of her fan and heater. On the basis of these rules, our system can provide a suggestion to the user when the environmental context in the user's office changes (e.g., if the temperature rises above a given threshold value). It is important to note that, whenever, the system suggests an adaptation (e.g., "Shall I turn on the fan?"), the user can dismiss the suggestion.

An actuator based on X10 (X10, 2004) technology is used to turn on/off the user's heater, fan or lamp. An early version of the system required the user to 'imagine' that the system was capable of turning on the fan when the physical context conditions were met. Again, we found that this approach was far from suitable for enabling us to gain an appreciation of the real reaction that a user would have to the system actually turning on her fan and highlights the importance of using working prototypes when developing this kind of interactive ubicomp application.

Our system comprises a database with several tables, storing raw context history, symbolised context history and both learnt and user-specified rules. As illustrated in Figure 2, the system also consists of a context manager, an inference engine, and an adaptation manager.

The context manager collects context from sensors (Byun and Cheverst, 2004). If within a period of one minute there is at least one change in a raw context value

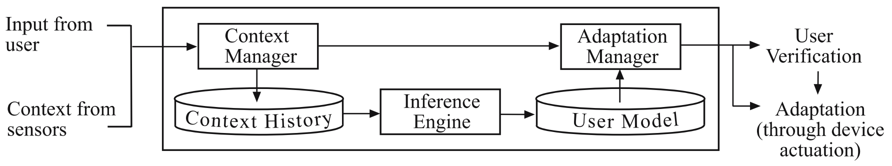


Figure 2. The design for providing dynamic adaptations.

(e.g., if the temperature changes from 24 to 25 °C, or the user turns on/off the fan), then the context manager generates a "context changed" event and stores the context in the context history. That is, while no state changes occur, the size of stored context history remains constant. The context manager also performs the symbolisation of raw context values.

The adaptation manager listens for the "context changed" events generated by the context manager. Next, the adaptation manager decides which adaptation must be made on the basis of both the current context (e.g., the current temperature) and the user model (which contains the set of learnt or predefined rules that represent the user's preferences).

The inference engine is responsible for extracting rules based on the context history and placing these in the user model.

3.2. SELECTION OF A DECISION-TREE-BASED LEARNING ALGORITHM

In the research field of user modelling, machine learning is actively investigated as a practical method to learn a user's interests, preferences, knowledge, goals, habits, and/or other properties in order to adapt the services to the user's individual characteristics (see, for example Mitchell et al., 1994; Pohl, 1996). As was first described in the introduction section, our primary requirement for our system was that it should facilitate comprehensibility in order for the user to be able to understand what is being done by the system and why the system is doing it (i.e., to scrutinise the behaviour of the system). This overriding system requirement implied selecting a learning algorithm that could provide the user with an explicit and understandable explanation for proactive behaviour. Generating rules that are readily intelligible by humans is one of the advantages of decision tree learning (Mitchell et al., 1994). Thus, at an early design stage, we chose to adopt a decision tree learning approach (Byun and Cheverst, 2003).

3.3. USE OF FUZZY LOGIC

While a decision tree is accessible to an average human user, the underlying model of crisp cut points (see the upper image in Figure 3) does not quite match human thinking. To alleviate this problem without sacrificing scrutability of the inference process, we chose to use fuzzy logic. Instead of crisp boundaries between categories, fuzzy logic introduces a *membership function*, which reflects how well a given value falls into a category (see the lower image in Figure 3). On the basis of a fuzzy representation of context (Mantyjarvi and Seppanen, 2002), we use fuzzy decision trees (Janikow, 1996) for inference (see also Zeidler and Schlosser, 1996; Shiu et al., 2000; Guetova et al., 2002). Our rationale for the use of a fuzzy decision tree is its potential to express symbolically the rules governing the system's proactive behaviour. In turn, this representation supports users' understanding of these rules, which provides a basis for scrutinising system behaviour. Enabling

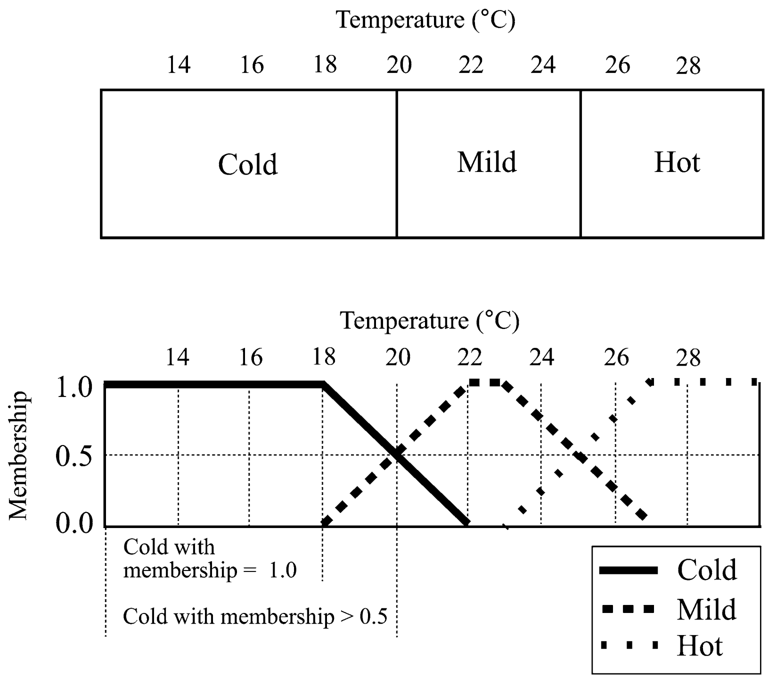


Figure 3. Conventional (top) and fuzzy (bottom) representations of temperature.

users' access to a symbolic representation of these rules (i.e. rule visualisation) supports users in developing both a structural ("how the system works") and a functional ("how the system can be used") model of the system.

Figure 3 illustrates conventional and fuzzy representations of temperature. Using the example partitions shown in this figure, the temperature 26 °C would be discretised into the categorical value 'hot' with the membership value 0.75, and the temperature 28 °C would be discretised into the same categorical value but with the membership value 1. Consequently, even though the two temperatures 26 and 28 °C have the same categorical value 'hot', they have different membership values in accordance with people's natural way of understanding such categorisations. Therefore, we would argue that this fuzzy representation improves the communication between users and our system. Note that the cut points shown in Figure 3 are for illustration only and that the system we present later in this article provides a means for the user to specify their own cut points if they so desire.

The membership functions of the fuzzy sets, in conjunction with information gains, are used to build a fuzzy decision tree. We will illustrate this process using the small set of context history (in the form of continuous values) shown in Table I (note that this same history is also used in the user interface walkthrough presented in Section 5). Table II shows the discretised set of context history based on Table I and the cut points for each attribute that is defined in our prototype system (the actual cut points used are those shown in Figure 15). For example, in the

Table I. A continuous-valued raw context history

Nr.	Date	Time	Temp	Noiselevel	Humidity	Light	Window	Fan	Heater
1	2004-26-11	14:36:01	23	55	30	52	Closed	Off	Off
2	2004-26-11	14:37:01	24	55	30	49	Closed	On	Off
3	2004-26-11	14:38:01	25	55	30	51	Closed	On	Off
4	2004-26-11	14:39:01	26	55	30	50	Closed	On	Off
5	2004-26-11	14:40:01	22	68	30	50	Closed	Off	Off
6	2004-26-11	14:41:01	22	62	30	50	Closed	Off	Off
7	2004-26-11	14:42:01	21	55	30	49	Closed	Off	On
8	2004-26-11	14:43:01	20	55	30	50	Closed	Off	On
9	2004-26-11	14:44:01	18	55	30	50	Closed	Off	On
10	2004-26-11	14:45:01	19	76	30	50	Closed	Off	On

Table II. A discretised context history including membership details of the Fan rule

Nr.	Date	Time	Temp	Noise Level	Humidity	Light	Window	Fan	Heater	Membership
1	2004-26-11	14:36:01	Mild	Loud	Normal	Normal	Closed	Off	Off	0.6
2	2004-26-11	14:37:01	Mild	Loud	Normal	Normal	Closed	On	Off	0.45
3	2004-26-11	14:38:01	Hot	Loud	Normal	Normal	Closed	On	Off	0.3
4	2004-26-11	14:39:01	Hot	Loud	Normal	Normal	Closed	On	Off	0.45
5	2004-26-11	14:40:01	Mild	Loud	Normal	Normal	Closed	Off	Off	0.6
6	2004-26-11	14:41:01	Mild	Loud	Normal	Normal	Closed	Off	Off	0.6
7	2004-26-11	14:42:01	Mild	Loud	Normal	Normal	Closed	Off	On	0.45
8	2004-26-11	14:43:01	Mild	Loud	Normal	Normal	Closed	Off	On	0.3
9	2004-26-11	14:44:01	Cold	Loud	Normal	Normal	Closed	Off	On	0.6
10	2004-26-11	14:45:01	Cold	Loud	Normal	Normal	Closed	Off	On	0.45

case of the first row, the temperature 23 °C is discretised to 'mild' with membership 1 on the basis of the cut points in Figure 15. Other continuous-valued attributes can be discretised in the same manner:

NoiseLevel is 'loud' with membership 1,
Humidity is 'normal' with membership 0.6,
Light is 'normal' with membership 1.

In the case of the Window attribute, the membership value of 'closed' is considered to be 1, because it is not a continuous-valued attribute. In order to capture the overall extent to which the predictive attributes have the specified values, we define a membership value for the rule as a whole (see Table II) in terms of the product of the membership values of all of the attributes on the left-hand side:

$$\begin{aligned}&\chi_{\text{Temp}}(\text{mild})\,\chi_{\text{NoiseLevel}}(\text{loud})\,\chi_{\text{Humidity}}(\text{normal})\,\chi_{\text{Light}}(\text{normal})\,\chi_{\text{Window}}(\text{closed})\\&= 1.0 \cdot 1.0 \cdot 0.6 \cdot 1.0 \cdot 1.0\\&= 0.6\end{aligned}$$

3.3.1. *Calculating the Root Node for the Fuzzy Tree with the Fan target function*

The first step in building a fuzzy decision tree is finding the root node. For this purpose, the information gain for each attribute *A* must be calculated. In the following calculation, *S* refers to the whole set of context history and the base of the logarithm is 2. The formulas for the calculation of entropy and information gain are essentially the same as those for building a conventional decision tree. However, in the case of a fuzzy decision tree, the membership values of the target attribute (for example, the first row membership of Fan is 0.6) should be used in the calculation. The corresponding formulas for the entropy *E(S)* and the information gain *G(S, A)* are as follows:

$$E(S) = -\frac{m_{\text{on}}}{m_{\text{Fan}}}\log\frac{m_{\text{on}}}{m_{\text{Fan}}} - \frac{m_{\text{off}}}{m_{\text{Fan}}}\log\frac{m_{\text{off}}}{m_{\text{Fan}}}$$

where

$$m_{\text{on}} = \sum_{s\in S}\chi_{\text{Fan}}(\text{on}) \quad \text{sum of all membership values for Fan} = \text{on}$$

$$m_{\text{off}} = \sum_{s\in S}\chi_{\text{Fan}}(\text{off}) \quad \text{sum of all membership values for Fan} = \text{off}$$

$$m_{\text{Fan}} = m_{\text{on}} + m_{\text{off}}$$

$$G(S, A) = E(S) - \frac{m_x}{m_A}E(S, A = x) - \frac{m_y}{m_A}E(S, A = y) - \frac{m_z}{m_A}E(S, A = z)$$

where

$$m_x = \sum_{s\in S}\chi_A(x) \quad \text{sum of all membership values for } A = x$$

$$m_y = \sum_{s\in S}\chi_A(y) \quad \text{sum of all membership values for } A = y$$

$$m_z = \sum_{s\in S}\chi_A(z) \quad \text{sum of all membership values for } A = z$$

$$m_A = m_x + m_y + m_z$$

Therefore, the entropy of the whole set and the information gain of Temp can be calculated from the membership values shown in the last column of Table II as follows:

$$m_{\text{on}} = 0.45 + 0.3 + 0.45 = 1.2$$

$$m_{\text{off}} = 0.6 + 0.6 + 0.6 + 0.45 + 0.3. + 0.6 + 0.45 = 3.6$$

$$E(S) = -\frac{1.2}{4.8}\log\frac{1.2}{4.8} - \frac{3.6}{4.8}\log\frac{3.6}{4.8} = 0.813$$

$$E\,(S, \text{Temp} = \text{hot}) = -\frac{0.3+0.45}{0.3+0.45}\log\frac{0.3+0.45}{0.3+0.45} - \frac{0}{0.3+0.45}\log\frac{0}{0.3+0.45} = 0$$

$$
\begin{aligned}
E\,(S,\text{Temp} = \text{mild}) &= -\frac{0.45}{\underbrace{0.6 + 0.45 + 0.6 + 0.6 + 0.45 + 0.3}_{=3.0}} \log \frac{0.45}{3.0} \\
&\quad - \frac{2.55}{3.0} \log \frac{2.55}{3.0} = 0.61 \\
E\,(S,\text{Temp} = \text{cold}) &= -\frac{0}{0.6 + 0.45} \log \frac{0}{0.6 + 0.45} \\
&\quad - \frac{1.05}{0.6 + 0.45} \log \frac{1.05}{0.6 + 0.45} = 0 \\
G(S, A) &= E(S) - \frac{0.75}{4.8} E(S, \text{Temp} = \text{hot}) - \frac{3.0}{4.8} E(S, \text{Temp} = \text{mild}) \\
&\quad - \frac{1.05}{4.8} E\,(S, \text{Temp} = \text{cold}) \\
&= 0.43
\end{aligned}
$$

In the case of the attributes NoiseLevel, Humidity, Light, and Window, all information gains are zero since they have only one attribute value (for example, all values of NoiseLevel are 'loud'). According to the information gain measure, Temp would provide the best prediction for target attribute Fan. Therefore it is selected as the root node.

Now let us consider a subset of the context history that contains the cases where temperature is mild (row numbers 1, 2, 5, 6, 7, 8). In this case, the information gains of all remaining attributes are also zero, because they have only one attribute value. Therefore, we can create a leaf node here. The membership values of 'on' and 'off' in the leaf node can be calculated by normalizing the target membership values in the subset: the membership of 'on' is:

$$
\frac{m_{\text{on}}^{\text{Temp}=\text{mild}}}{m^{\text{Temp}=\text{mild}}} = \frac{0.45}{0.6 + 0.45 + 0.6 + 0.6 + 0.45 + 0.3} = 0.15
$$

$$
\frac{m_{\text{off}}^{\text{Temp}=\text{mild}}}{m^{\text{Temp}=\text{mild}}} = \frac{0.6 + 0.6 + 0.6 + 0.45 + 0.3}{0.6 + 0.45 + 0.6 + 0.6 + 0.45 + 0.3} = 0.85
$$

where

$$
\begin{aligned}
m_{\text{on}}^{\text{Temp}=\text{mild}} &= \sum_{s \in S|\text{Temp}=\text{mild}} \chi_{\text{Fan}}(on) \quad \text{sum of all membership values for Fan} = \text{on} \\
m_{\text{off}}^{\text{Temp}=\text{mild}} &= \sum_{s \in S|\text{Temp}=\text{mild}} \chi_{\text{Fan}}(off) \quad \text{sum of all membership values for Fan} = \text{off} \\
m^{\text{Temp}=\text{mild}} &= m_{\text{on}}^{\text{Temp}=\text{mild}} + m_{\text{off}}^{\text{Temp}=\text{mild}}
\end{aligned}
$$

Since the membership value of 'off' for this node is greater than the one for 'on', we can extract the rule 'if Temp = mild and NoiseLevel = loud and Humidity = normal and Light = normal and Window = close, then Fan = off with membership 0.85'.

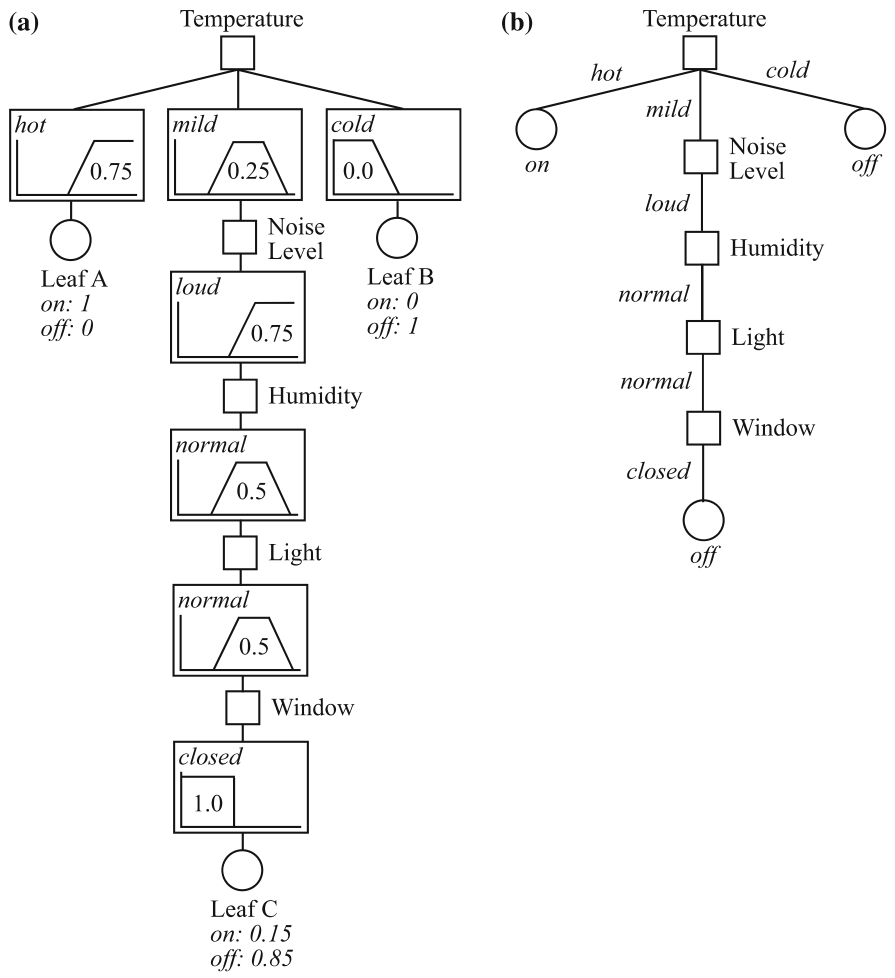


Figure 4. (a) A fuzzy decision tree for Fan (see section 3.3.1. for justification of values shown in nodes). (b) A conventional decision tree for Fan.

In the cases of the subset of context history for Temp = 'hot' and Temp = 'cold', the information gains of all remaining attributes are zero for the same reason as in the above case, and all target values are the same ('off'). Therefore, we can create leaf nodes here without normalizing the target membership values. The rules from these two cases are: 'If Temp = hot, then Fan = on with membership 1' and 'If Temp = cold, then Fan = off with membership 1'. Now we can draw a fuzzy decision tree as shown in Figure 4a. In contrast, if a new nonfuzzy decision tree algorithm were used, the tree produced would appear as that shown in Figure 4b. Please note that the peculiar shape of the decision trees is a result of the small size of the dataset used in the example. The trees generated by the actual system are more complex, having a higher branching factor on each level.

3.3.2. *Illustration of Adaptation Procedure based on the Fuzzy Tree with the Fan Target Attribute*

In this subsection, we present an actual example for how adaptation is performed by our prototype system. The example is based on the fuzzy decision tree shown in Figure 4a, which was derived from the small sample of context history shown in Table I. Let us assume that the environmental conditions in the user's office are as follows:

Temperature is 26 °C, NoiseLevel is 55 dBA, Humidity is 29%, Light is 68 lux, and Window is closed.

The first step of adaptation in this new situation is the fuzzification of each context attribute. The corresponding membership functions are shown in the boxes between the nodes in the fuzzy decision tree shown in Figure 4a. For example, according to the box shown just below the 'NoiseLevel' node, 55 dBA would be categorised as 'loud' with membership 1. We can then compute the value of the membership function χ for each value of the target attribute Fan in each leaf node (A, B, C):

Leaf A:

$$\chi\,(\text{Fan} = \text{on}) = \chi_A(\text{Fan} = \text{on})\chi(\text{Temp} = \text{hot}) = 1 \cdot 0.75 = 0.75$$
$$\chi\,(\text{Fan} = \text{off}) = \chi_A(\text{Fan} = \text{off})\chi(\text{Temp} = \text{hot}) = 0 \cdot 0.75 = 0$$

Leaf B:

$$\chi\,(\text{Fan} = \text{on}) = \chi_B(\text{Fan} = \text{on})\chi(\text{Temp} = \text{cold}) = 0 \cdot 0 = 0$$
$$\chi\,(\text{Fan} = \text{off}) = \chi_B(\text{Fan} = \text{off})\chi(\text{Temp} = \text{cold}) = 1 \cdot 0 = 0$$

Leaf C:

$$\begin{aligned}\chi\,(\text{Fan} = \text{on}) &= \chi_C(\text{Fan} = \text{on})\chi(\text{Temp} = \text{mild})\chi(\text{NoiseLevel} = \text{loud})\\ &\quad \cdot\chi\,(\text{Humidity} = \text{normal})\,\chi\,(\text{Light} = \text{normal})\chi(\text{Window} = \text{closed})\\ &= 0.15 \cdot 0.25 \cdot 1.0 \cdot 0.5 \cdot 0.5 \cdot 1.0 = 0.01\\ \chi\,(\text{Fan=off}) &= \chi_C(\text{Fan} = \text{off})\chi(\text{Temp} = \text{mild})\chi(\text{NoiseLevel} = \text{loud})\\ &\quad \cdot\chi\,(\text{Humidity} = \text{normal})\,\chi\,(\text{Light} = \text{normal})\chi(\text{Window} = \text{closed})\\ &= 0.85 \cdot 0.25 \cdot 1.0 \cdot 0.5 \cdot 0.5 \cdot 1.0 = 0.05\end{aligned}$$

Therefore, the total membership values for 'Fan = on' and 'Fan = off' are:

$$\chi(\text{Fan} = \text{on}) = \sum_{A,B,C} \chi(\text{Fan} = \text{on}) = 0.75 + 0 + 0.01 = 0.76$$
$$\chi(\text{Fan} = \text{off}) = \sum_{A,B,C} \chi(\text{Fan} = \text{off}) = 0 + 0 + 0.05 = 0.05$$

Note that each leaf node (the categorical value of the target function: on and off) has a membership value that stems from the membership values of context history data. In order to choose the value of the target function, the membership values of leaf nodes and the membership values of current contexts are multiplied and added for the same target values. Finally, the target value that has the largest total is selected as a suggestion, i.e. 'on' with certainty level 0.76 for this proactive service. Therefore, our system would turn on the fan (or suggest turning on the fan, depending on the user's preferences) with a confidence level of 0.76 (provided that this value was higher than the threshold that determines whether proactive behaviour should occur).

3.3.3. *The Fluctuation Problem*

In any control system that controls part of an environment populated by human users, fluctuations around a cut point pose a problem. For example, if a fan is turned on whenever the temperature is higher than 25°, it might happen that the system continuously turns the fan off and on if the temperature fluctuates around 25°. While fuzzy logic does not eliminate those problems *per se*, it is possible to apply certain strategies to address fluctuations.

Perhaps the simplest approach to introduce a further action "leave unchanged" for the right-hand side of rules. For example, if the rules specify that the fan is to be turned off under 24°, to be turned on above 26°, and to be left unchanged otherwise, fluctuation on the basis of small temperature changes cannot occur. If, for example, the fan is on because the temperature is 27 degrees, it will not be turned off again until the temperature drops below 24°.

In another approach, the time interval since the last change in the state of the device could be made a feature to be considered in the left-hand side of the rules. In that case, for example, the system might learn not to make any change sooner than 5 min after the previous change.

3.4. EVALUATION METHODOLOGY

Given our interest in both the technical and human factors issues associated with the development of a system supporting proactive behaviour in an intelligent office environment, our evaluation methodology has been to start our exploration with the development of a simplified but working prototype system that can be deployed at a relatively early stage. This early deployment has enabled user studies to commence and provide valuable feedback with regard to user acceptance of the IOS concept and to drive future developments; in this respect we are adopting a user-centred design approach. In order to accelerate exploration of issues where appropriate, our evaluation methodology also includes the use of questionnaires to focus on specific issues that are revealed by user feedback from the prototype. For example, after the use of the prototype revealed issues about how users would prefer to visualise inferred rules and the confidence values associated with those rules when scrutinising the behaviour of the system, a questionnaire was used to enable a relatively quick and low-cost investigation into this issue (the results of this particular aspect of our investigation are described in Section 6.2.3.2). However, as is argued below, the danger with this approach is that such responses are not provided 'in situ' and that therefore such results must be judged with a certain degree of scepticism. Our intention is for our current user studies to develop into longtitudinal studies that should provide much-needed insights into the longer-term implications of using and maintaining the IOS. Given the relatively short-term nature of our studies, so far we can claim only to provide formative results that may point to directions for further study.

4. The Hardware and Software Components of the Intelligent Office System

4.1. HARDWARE COMPONENTS

The hardware components of the system are required for sensing physical context in the user's office and for actuating supported devices. Actuation (e.g., turning a device on or off) occurs if a rule learnt by the system is triggered by the current context (and the user has specified that actuation can occur without user verification) or if the user chooses to use the system's graphical user interface to instruct the system to turn a given device on or off. The main hardware components of the system comprise two different sensors (the 'off-the-shelf' DrDaq data logger and a purpose build current sensor) and the actuation system (based on the X10 system). The following subsections describe these components in more detail.

4.1.1. *The DrDaq Data Logger*

In our experiment, we use a DrDAQ data logger from Pico Technology Ltd (see Figure 5) in order to obtain the state of the user's office environment. The DrDAQ data logger has built-in sensors for detecting various environmental contexts (e.g., light level, sound level and temperature) and two external sensors. One of these external sensors is used to sense whether or not the user's window is currently open, while the other one is used to sense humidity.

It is important to note that the accuracy of temperature sensed by the DrDAQ is 2 °C at around 25 °C (the error margin depends on the temperature). If the threshold between 'hot' and 'mild' is 25 °C, then this 2 °C error margin could clearly be very significant (e.g., the same true temperature could be sensed as either 'hot' or 'mild'). One approach we plan to investigate is the setting of appropriate thresholds to determine the change required in a given raw sensor value before a new value is recorded in the context history. This problem is similar in nature to the problem of reacting to apparent changes in location sensed by Geographical Positioning System (GPS) in which the inaccuracy can be in the region of 10 m.

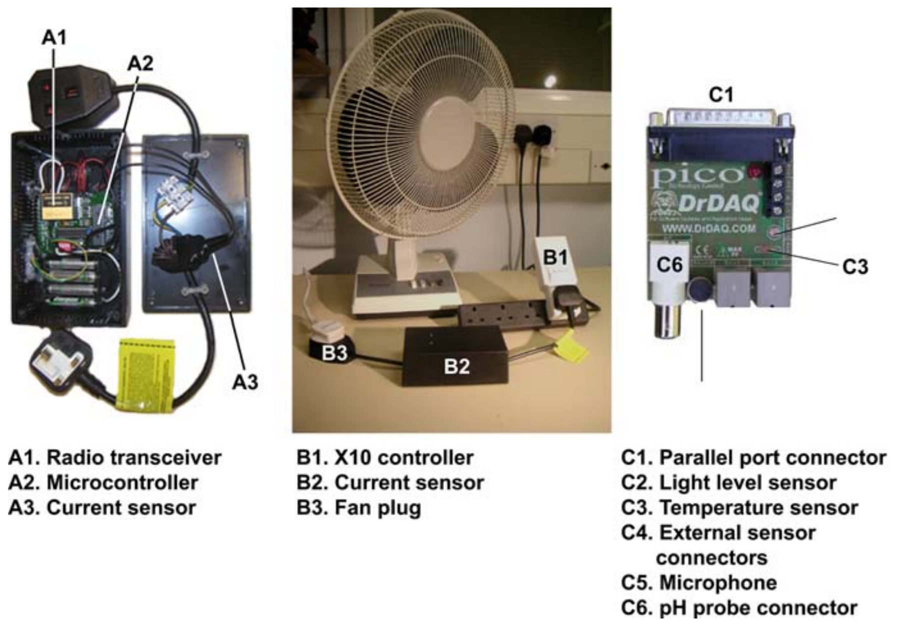


Figure 5. The Current Sensor (left), the Current Sensor attached to a user's fan (center) and the DrDaq Data Logger (right).

As was mentioned previously, the DrDaq module is used for detecting the state of the window. This task is accomplished with an external reed switch sensor that is positioned on the window frame and a small magnet located on the moving part of window. When the window is closed, the magnet sits very close to the reed switch, causing it to open, and when the window is opened, the magnet moves away from the reed switch, causing it to close – these changes are recognised by the DrDaq module and sent to the Intelligent Office System.

4.1.2. *X10-Based Actuation*

In order to achieve actuation of devices, we have utilised X10 technology. It is not possible to send X10 signals through the internal power wiring of the university Computing Department in which our experiments have been taking place, so a trivial but highly practical solution has simply been to plug the fan and X10 controller into the same gangway extension (see Figure 5). Using this approach, the X10-based control signals need only travel successfully along the length of the gangway extension.

Unfortunately, the reliability of standard X10 signalling cannot be guaranteed or even verified in many cases, so to be sure of the status of a device, we have developed the *current sensor* (i.e., the sensor for detecting changes in electrical current) described in the following section.

4.1.3. *Current Sensor*

Early user trials revealed that typical users tend to be reluctant to click manually on a check box in a dialog box every time they change the state of the fan in their office in order to update their context history. Consequently we set about developing a system that would enable the action of turning on a device to be automatically detected. The current sensor that we developed (specifically for the deployment of the IOS) is able to detect whether the fan, or any other standard electrical appliance connected to the current sensor, is switched on or off at any given time. It does this by measuring the amount of current that passes through the mains power cable. A magnetic coil wound around the mains line produces a voltage whenever the appliance is turned on, which is proportional to the amount of current drawn by the appliance. This voltage is continuously sampled using an analogue-to-digital (A/D) converter and a PIC microcontroller. Software running on the PIC then detects transitions between the 'ON' and 'OFF' states of the appliance on the basis of the output of the A/D converter. The current sensor in Figure 5 is based on a Smart-Its (Holmquist et al., 2001) prototyping device. It provides an RS232 and a wireless radio interface to other devices. This arrangement allows direct interfacing with the IOS running on the PC.

4.2. SOFTWARE COMPONENTS

The database server used is the open source product MySQL® (www.mysql.com). In addition to MySQL's excellent reliability and ease of use, its open source nature enables the entire IOS software to be made available for experimentation by other research groups free of charge.

Referring back to the overall architecture diagram illustrated in Figure 2, the 'Context Manager', 'Inference Engine' and 'Adaptation Manager' components of the software are all written in Java and are designed to be compiled and run under any Java 2 environment. Additionally, MySQL provides an official Java DataBase Connectivity (JDBC) driver, which was used throughout the system.

A significant portion of the software code is user interface code. The following section describes in detail the user interaction supported by the system through its graphical user interface.

5. User Interface Walkthrough

In order to help illustrate the behaviour of the system, we have generated rules that are consistent with a sample piece of context history. We will now show what behaviour would be triggered on the basis of hypothetical environment values such as temperature.

The current GUI is based on a main control window that is designed to run on a display that is separate from the user's primary desktop display (or displays), as is shown below in Figure 6.

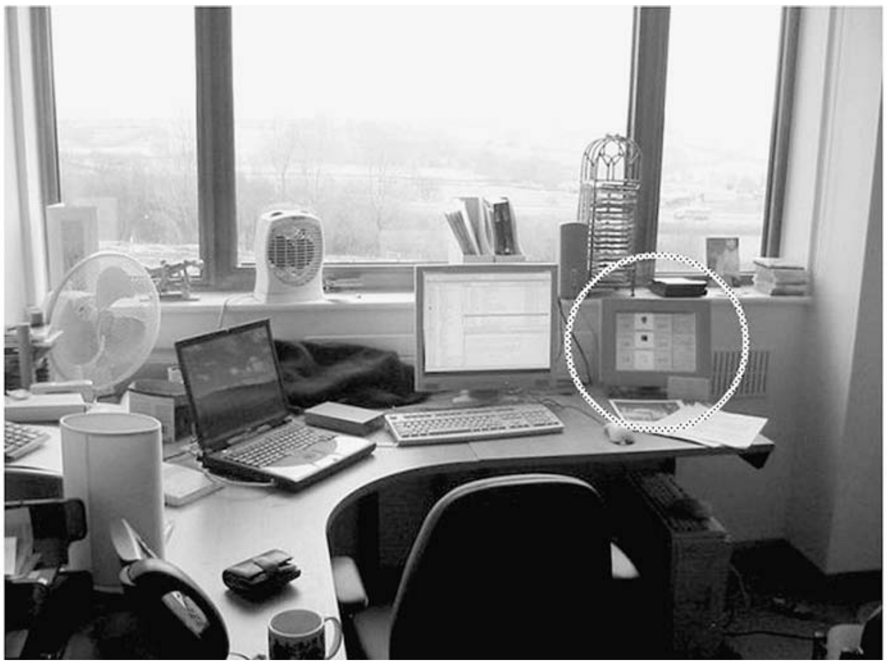
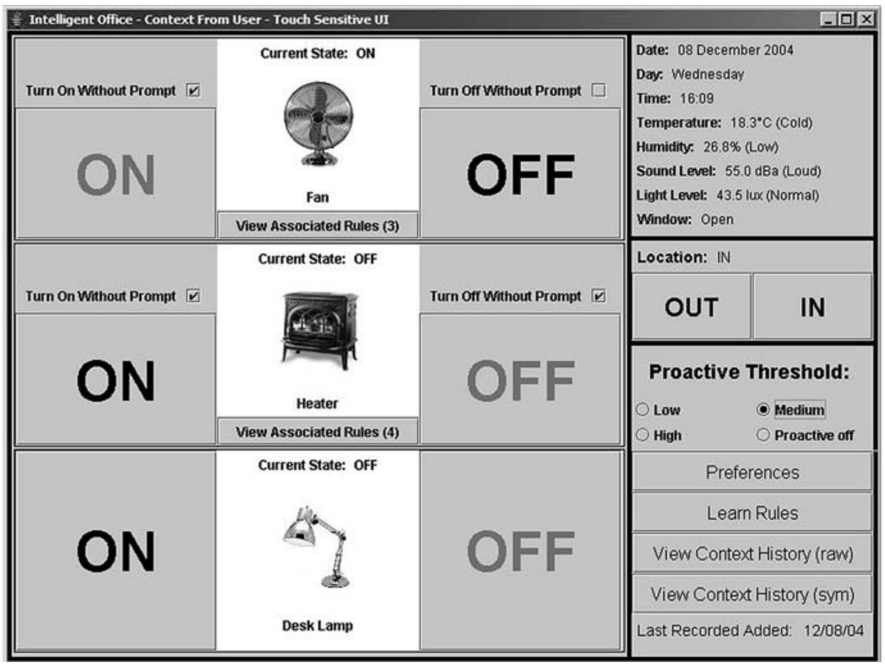


Figure 6. The typical deployment of the control system (on a separate display).

The Main Control window to the IOS is shown in Figure 7. This GUI is divided into four parts which correspond to different aspects of the functionality. The *Device Control* part (in the left-hand two-thirds of the window) enables the user to turn on and off devices in the office, to set preferences for fully automated behaviour, and to view associated rules. The current state of a device is shown through text and through greying out of the text on the button that is not needed. The *Current State of Environment* part (top right-hand part of window) shows the values currently being read from the DrDaq sensor and also presents these values in terms of their main fuzzy classification (in order to help reinforce the user's understanding of the relationship between numeric values and classifications). An earlier version of the software did not contain this information, and it was only added following requests from a number of users.

The *Setting Location* part (the centre right-hand part of the window) is the one place where the system still does require the user to inform the system manually of a context event (i.e., whether the user is in the office or not); we are currently investigating a range of techniques for automatically sensing this information with the high degree of accuracy required. Finally, the *Intelligent Office Functions* part (the bottom right-hand part of window) is the primary part of the GUI for enabling the user to control the behaviour of the IOS, including the ability to set the proactive threshold to an appropriate level (including 'off') and to view the context history.


Intelligent Office - Context From User - Touch Sensitive UI

Current State: ON
Turn On Without Prompt ☑
Turn Off Without Prompt ☐
Fan

Current State: OFF
Turn On Without Prompt ☑
Turn Off Without Prompt ☑
Heater

Current State: OFF
Desk Lamp

Date: 08 December 2004
Day: Wednesday
Time: 16:09
Temperature: 18.3°C (Cold)
Humidity: 26.8% (Low)
Sound Level: 55.0 dBa (Loud)
Light Level: 43.5 lux (Normal)
Window: Open

Location: IN

Proactive Threshold: ○ Low [● Medium] ○ High ○ Proactive off

Last Recorded Added: 12/08/04
ON
OFF
View Associated Rules (3)
ON
OFF
View Associated Rules (4)
ON
OFF
OUT
IN
Preferences
Learn Rules
View Context History (raw)
View Context History (sym)

Figure 7. The Main Control GUI.

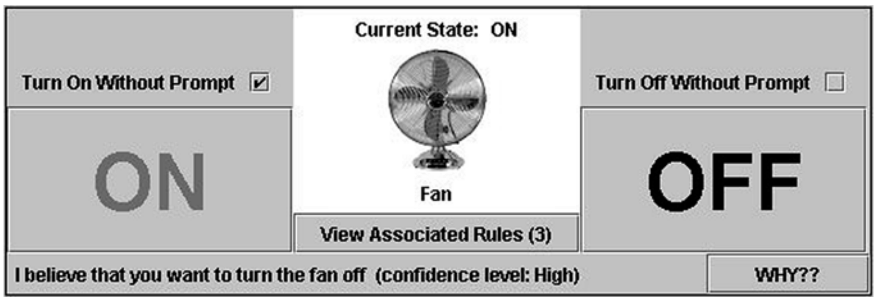
The following two sections describe the *Device Control* and *Intelligent Office Functions* parts of the main GUI in more detail.

5.1. DEVICE CONTROL

The *Device Control* part of the window is used to enable the user to turn on or off devices and it is intended to be the primary means for users to control devices. We have found this part of the window to be much used in practice, largely due to the convenient placement of the touch screen display (as shown in Figure 6). A major benefit of this approach is that context relating to the turning on and off of devices can be automatically captured and stored in the context history.

5.2. WHEN A RULE FOR DEVICE ACTUATION IS TRIGGERED

When a suggestion prompt is issued (which occurs if the user has indicated that a prompt rather than automatic action is required) it is displayed on the main control GUI. For example, if the system suggests that the fan should be turned off, then the UI changes to that shown below in Figure 8 and the text on the 'OFF' button flashes black and white. Note that in this example, the user is shown the confidence level of the rule as a categorical value, in this case 'High' (see Section 6.2.3.3 for questionnaire results regarding user's preferences for visualising confidence levels).



I believe that you want to turn the fan off (confidence level: High)

Figure 8. Display of a system prompt.

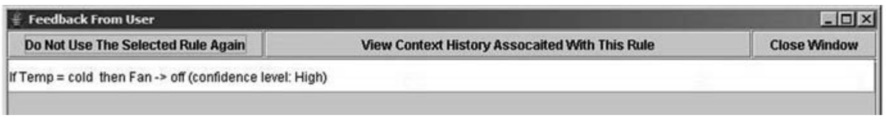
However, by altering their preferences a user can also choose to view the confidence level as a decimal number, e.g. 0.85, if she prefers.

The user will not always want the system to offer a prompt before turning on a device. For this reason, the GUI presents the user with tick boxes for allowing proactive actions to occur automatically. Given the selection of tick boxes shown in Figure 8, if a rule (with a high confidence level as per the user's preferences) became triggered for turning on the user's fan, the fan would automatically be turned on and the system would display the following message: "I have just turned on the fan for you (confidence level: High)". This facility was incorporated on the basis of feedback from users of the system and the results of the questionnaire study (see Section 6.2.3.1).

If the user wishes to enquire why that suggestion was made (i.e. scrutinise the system), she can press on the appropriate button in order to view a window such as the one shown below in Figure 9. Please note that for the purposes of this walk-through we are using only a simple set of rules; in practice more complicated rules can be generated.

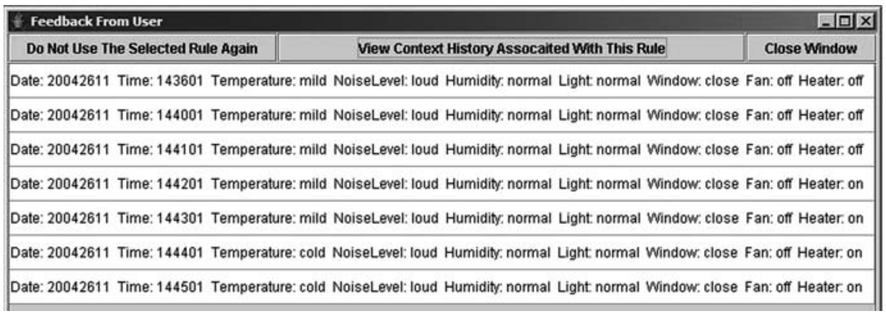
Note that at this point the user also has the ability to stop using the rule that generated the suggestion being used again by selecting the 'Do Not Use The Selected Rule Again' button. If the user wishes to scrutinise the actual context history associated with a given rule, she can do so by selecting the rule and clicking on the appropriate button. An example of the kind of output produced is shown in Figure 10.

The user can also choose to view all the rules associated with a given appliance at any time by pressing the 'View Associated Rules' button on the appropriate 'Device Control' part of the Main Control GUI. The rules associated with the


Feedback From User
Do Not Use The Selected Rule Again
View Context History Assocaited With This Rule
Close Window

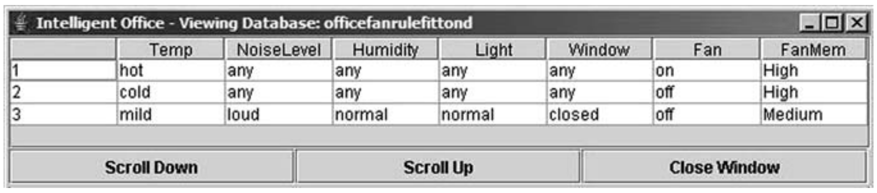
If Temp = cold then Fan -> off (confidence level: High)

Figure 9. Scrutinising the rule behind a prompt to turn off the fan.


Feedback From User
Do Not Use The Selected Rule Again
View Context History Assocaited With This Rule
Close Window

```
Date: 20042611  Time: 143601  Temperature: mild  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: off
Date: 20042611  Time: 144001  Temperature: mild  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: off
Date: 20042611  Time: 144101  Temperature: mild  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: off
Date: 20042611  Time: 144201  Temperature: mild  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: on
Date: 20042611  Time: 144301  Temperature: mild  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: on
Date: 20042611  Time: 144401  Temperature: cold  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: on
Date: 20042611  Time: 144501  Temperature: cold  NoiseLevel: loud  Humidity: normal  Light: normal  Window: close  Fan: off  Heater: on
```

Figure 10. Scrutinising the context history associated with a given rule.



	Temp	NoiseLevel	Humidity	Light	Window	Fan	FanMem
1	hot	any	any	any	any	on	High
2	cold	any	any	any	any	off	High
3	mild	loud	normal	normal	closed	off	Medium

Figure 11. Scrutinising the rules associated with the fan.

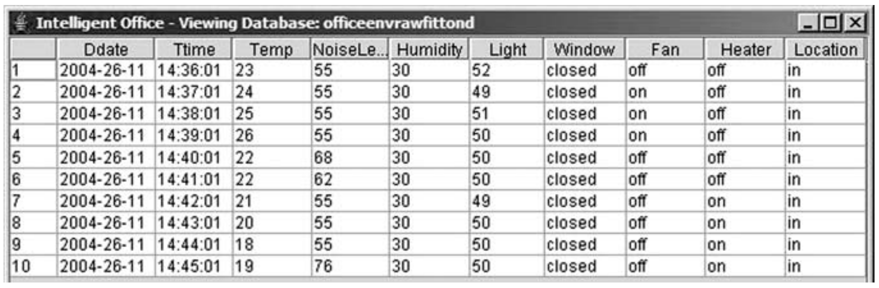
fan, which are based on the context history shown in Tables I and II, are shown in Figure 11. Note that the values shown in the 'FanMem' column in Figure 11 effectively provide a confidence level for each rule.

5.3. SETTING THE THRESHOLD FOR PROACTIVE BEHAVIOUR

The user is able to control the threshold at which proactive actions occur by clicking on the appropriate radio button. For example, if the user wishes actions with a confidence level less than High to be ignored, then she would click on the High setting. This function was added on the basis of the feedback from users testing the system in their offices and the results of the questionnaire study. The user can also turn off any proactive behaviour using this part of the GUI.

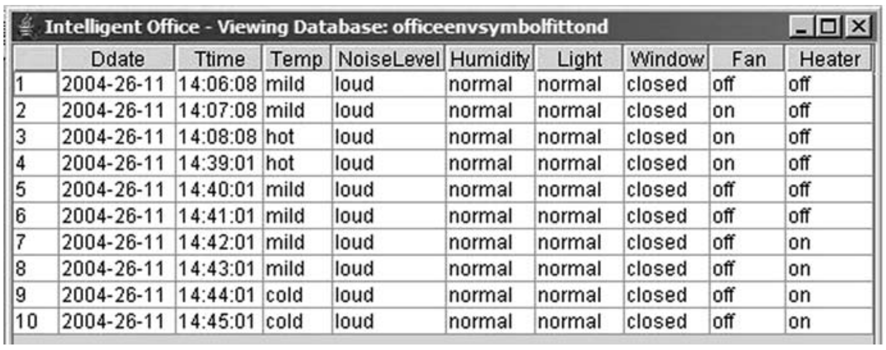
5.4. VIEWING THE CONTEXT HISTORY

As was described in Section 2, fundamental to our approach is the notion of enabling the user to understand (or at least be able to scrutinise) the rules behind the system's behaviour. For this reason, and on the basis of the results of the questionnaire, we have enabled the user to view her context history. The user currently has two choices for viewing context history, either as raw data with numerical values, as shown in Figure 12, or, symbolic data with symbolic values, as shown in Figure 13.


Intelligent Office - Viewing Database: officeenvrawfittond

	Ddate	Ttime	Temp	NoiseLe..	Humidity	Light	Window	Fan	Heater	Location
1	2004-26-11	14:36:01	23	55	30	52	closed	off	off	in
2	2004-26-11	14:37:01	24	55	30	49	closed	on	off	in
3	2004-26-11	14:38:01	25	55	30	51	closed	on	off	in
4	2004-26-11	14:39:01	26	55	30	50	closed	on	off	in
5	2004-26-11	14:40:01	22	68	30	50	closed	off	off	in
6	2004-26-11	14:41:01	22	62	30	50	closed	off	off	in
7	2004-26-11	14:42:01	21	55	30	49	closed	off	on	in
8	2004-26-11	14:43:01	20	55	30	50	closed	off	on	in
9	2004-26-11	14:44:01	18	55	30	50	closed	off	on	in
10	2004-26-11	14:45:01	19	76	30	50	closed	off	on	in

Figure 12. Screenshot illustrating a 10 row sample of 'raw' context history.


Intelligent Office - Viewing Database: officeenvsymbolfittond

	Ddate	Ttime	Temp	NoiseLevel	Humidity	Light	Window	Fan	Heater
1	2004-26-11	14:06:08	mild	loud	normal	normal	closed	off	off
2	2004-26-11	14:07:08	mild	loud	normal	normal	closed	on	off
3	2004-26-11	14:08:08	hot	loud	normal	normal	closed	on	off
4	2004-26-11	14:39:01	hot	loud	normal	normal	closed	on	off
5	2004-26-11	14:40:01	mild	loud	normal	normal	closed	off	off
6	2004-26-11	14:41:01	mild	loud	normal	normal	closed	off	off
7	2004-26-11	14:42:01	mild	loud	normal	normal	closed	off	on
8	2004-26-11	14:43:01	mild	loud	normal	normal	closed	off	on
9	2004-26-11	14:44:01	cold	loud	normal	normal	closed	off	on
10	2004-26-11	14:45:01	cold	loud	normal	normal	closed	off	on

Figure 13. Screenshot illustrating a 10-row sample of 'symbolic' context history.

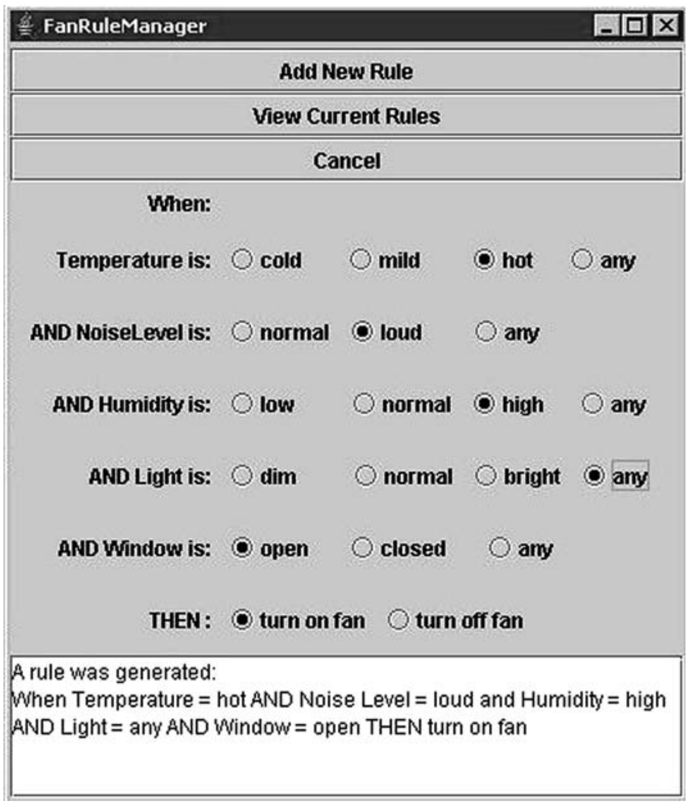
For reasons of simplicity, we have deliberately shown an amount of context history that is very small but large enough to convey the idea and to generate the rules shown in Figure 11.

5.5. SETTING PREFERENCES AND USER-DEFINED RULES

The IOS enables users to change a variety of settings or preferences (e.g. the boundaries used to separate different confidence categories used by the system). In addition, the system enables users to add their own rules (an example of which is shown in Figure 14) and to access and modify the 'cut-points' file, an example of which is shown in Figure 15.

5.6. INITIATING THE LEARNING PROCESS

By pressing the *Learn Rules* button on the Main Control GUI (shown in Figure 8) following the collection of a reasonable amount of context history, the user can request the system to start inferring rules from her context history. A few hours of context history can be sufficient; but the greater the size of context history used, the higher the system's ability to capture the nuances of the user's behaviour.



When:

Temperature is: cold, mild, [✓ hot], any

AND NoiseLevel is: normal, [✓ loud], any

AND Humidity is: low, normal, [✓ high], any

AND Light is: dim, normal, bright, [✓ any]

AND Window is: [✓ open], closed, any

THEN : [✓ turn on fan], turn off fan

A rule was generated:
When Temperature = hot AND Noise Level = loud and Humidity = high
AND Light = any AND Window = open THEN turn on fan

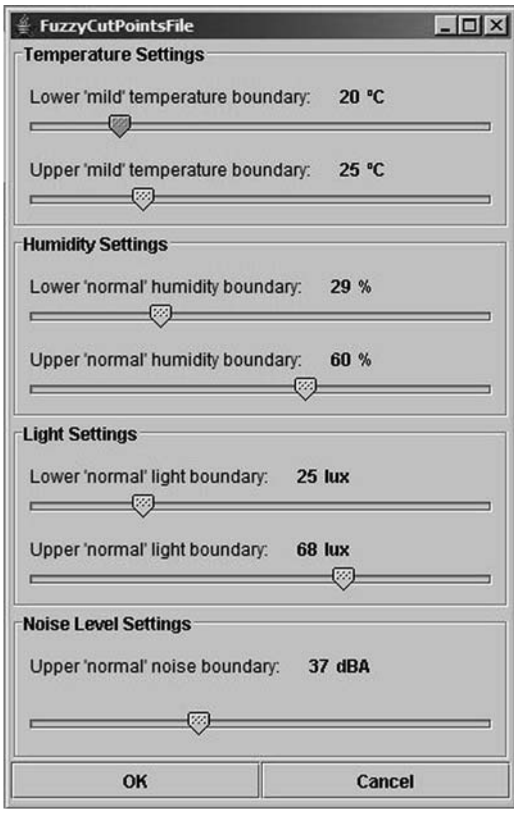
Figure 14. A user interface for enabling the user to add her own rules – the generated rule is shown at the bottom of the window.

At the design stage, we needed to consider whether to adopt an on-line or an off-line learning strategy. An on-line strategy would construct a user model incrementally in real-time, while an off-line strategy builds or updates the user model as a kind of batch job. Because of the high computational resources required to run an on-line learning system, the current version of the IOS utilises an off-line approach to learning.

Although we did not anticipate that the ability to scrutinise context history would be particularly popular with a great many nontechnical users, the results of the questionnaire (Section 6) indicated that many nontechnical users do desire such a facility. Furthermore, we believe that the user's trust in the system can be fostered with this *glass box* (Karlgren et al., 1994**)** approach to the user modelling aspects of the system.

6. Current State and Initial Evaluation

This section describes the current state of our prototype deployment (Section 6.1) and the results of our questionnaire-based study (Section 6.2).


FuzzyCutPointsFile

Temperature Settings

Lower 'mild' temperature boundary: 20 °C

Upper 'mild' temperature boundary: 25 °C

Humidity Settings

Lower 'normal' humidity boundary: 29 %

Upper 'normal' humidity boundary: 60 %

Light Settings

Lower 'normal' light boundary: 25 lux

Upper 'normal' light boundary: 68 lux

Noise Level Settings

Upper 'normal' noise boundary: 37 dBA

Figure 15. A user interface for enabling the user to access and modify the 'cut-points' file.

6.1. CURRENT STATE OF DEPLOYMENT

The initial version of the IOS was first deployed in February 2002. Since that time, a number of modifications and improvements to the system have been made as challenges to our approach have been encountered. Key modifications have been the introduction of a fuzzy logic approach in January 2003, the utilisation of X10 technology in March 2004, and a complete user interface modification in October 2004. In March 2005, the system underwent significant modification in order to increase overall reliability and ease of installation for nontechnical users. The software for the system is available on the web (at http://www.comp.lancs.ac.uk/computing/staff/kc/IntelligentOffice.html) so that other research groups can experiment with the IOS.

This actual IOS system itself was deployed in the offices of four of the developers for approximately one month, during which the IOS was subject to modification. As is described below, the insights of these four real users were used to inform the design of our questionnaire-based study, which involved participants with technical and nontechnical backgrounds and which revealed a generally positive approach toward proactive systems such as the IOS and provided a validation of the importance of enabling the user to scrutinise the reasons (i.e. rules) behind proactive behaviour.

6.2. AN INITIAL FORMATIVE QUESTIONNAIRE EVALUATION

The actual deployment of the system revealed many interesting issues, such as the need to reduce cost (in terms of perceived user effort) and maximise user benefit. It strongly informed the design of our questionnaire study, which sought both to explore the general attitude of participants towards the IOS and to investigate the following, more specific, issues:

- How much scrutability do users want and what are appropriate ways of supporting it?
- How much control do users want, and what should be automatic?

Consequently, the questionnaire study was primarily an explorative one, aiming to lead to hypotheses to be tested in subsequent evaluation studies.

6.2.1. *Questionnaire Design*

Table III presents a brief description of the various dimensions and associated variables which encode the concept of user's attitude with respect to an intelligent control system such as the IOS. Apart from measuring the general attitude towards such systems, the questionnaire also focuses on capturing two aspects of great interest for our work, namely control and scrutability.

The questionnaire contained 52 items with a 5-point Lickert scale and 13 open questions which enabled participants to make comments on the relevant issues. The user profile was captured through questions that called for standard factual data.

For the benefit of subjects who had not had experience with the IOS, a description of the system was given, along with examples of screen shots and scenarios describing how the system would behave under different circumstances.

6.2.2. *Participants*

The nonrandom sample comprised of 30 subjects: 26 potential users of the system and four actual users. Among them were 25 males and 5 females. 63% of the participants were younger than 35 years old. The 30 subjects differed greatly in the extent of their hands-on experience with systems like the IOS; the four subjects who had actually used the IOS represent one extreme, while the other extreme was represented by subjects who had no experience even with comparable systems. Indeed, while 40% of participants declared themselves to have had previous experience with a proactive system, 33% were not sure about what exactly constitutes a 'proactive system', a fact indicating that the concept of a proactive system bears some ambiguity. Completing the questionnaire required between twenty minutes and one hour.

Table III. Concept operationalisation regarding user attitudes towards an intelligent control system used as a basis for questionnaire design.

Dimensions	Variables	Items
General attitude towards intelligent control systems	Acceptance	Previous experience with such systems, (e.g., automatic doors, Microsoft Office Assistant) in different environments (i.e., home, work)
	Error tolerance	Tolerance of wrong decisions made by the system during/after the training phase; willingness to allow the system to learn by accepting system errors during learning
	Market potential	Intention to buy, estimated perceived value
Control and interruption	User control	Perceived importance of: ability to amend system decisions, system prompts before carrying out actions or without carrying out actions, ability to change between the system working automatically or with a prompt, availability for basic and expert views
Comprehensibility and scrutability	Visualisation (of rules)	Evaluation of: IF-THEN rule tables, text-based formulation of rules, decision trees, usage of colour within decision trees, context history
	Confidence level (associated with rules)	Preferred displayed format for confidence values, need to be able to choose the format, need to be able to set the confidence threshold for proactive actions and system prompts

6.2.3. *Results*

In the following discussion, we report the results of statistical tests in order to convey an idea of the likelihood that the results in question arose by chance alone. The specific probabilities reported should not be taken at face value, because of the large number of tests that were performed (including some whose results are not reported here because conventional levels of statistical significance were not attained). Instead, the results of the tests are intended to give an indication of which hypotheses deserve more systematic testing in the future.

The presentation of the evaluation study results is organised according to the dimensions and variables involved in the questionnaire design.

The majority of subjects reported being frustrated when a proactive system fails: for automatic doors 80% and for the Microsoft® Office assistant 50%.

More than 70% of subjects are happy to use technology at either home or work and feel at ease with it. However, the study revealed that participants differentiated between home and office environments when considering the acceptability of intelligent control systems. More specifically, over 66% of participants would be happy to have an intelligent control system deployed at home, while this figure drops to less than 50% for deployment in a work/office environment.

In a work environment, only 13% of participants agreed to let the system learn their behaviour patterns as a basis for its proactivity, when at home this changed to over 60%. This finding is related to the fact that within a work environment, people are less inclined to accept something which may impede their performance. The reliability of such systems, i.e. the Microsoft Office Assistant, is perceived as low. This attitude is related to participants' tolerance for system errors, which is generally low, decreasing from 50% during the training phase to 16% after that. Not surprisingly, there is a significant correlation between the degree of perceived usefulness of Microsoft Office Assistant and the willingness to tolerate system errors during the training phase ($r(28) = 0.38$, $p < 0.05$).

The perceived market value for such a system depends on system placement: 56% of study participants would pay more than 10 pounds for it when placed at home, while only 36% participants were willing to pay more than 10 pounds for an office-based system.

6.2.3.1. *Control and Interruptions.* More than 90% of participants in the study sample expressed the need for controlling the system (scoring 4 or 5 out of 5). Furthermore, 75% of participants desired the system to prompt before carrying out actions. However, 69% of participants wanted the ability to switch between the having the system working automatically or with a prompt. In terms of system transparency, 66% of participants wanted access to basic and expert views, the latter providing "... the ability to look at the 'rules' which have been generated".

Starting the system and forgetting about it contradicts the need of being in control so unsurprisingly it was an option that most participants did not approve of (17%).

6.2.3.2. *Rule Visualisation.* The questionnaire study also sought to explore the participant's preferences for different formats for visualising the rules embedded in the user mode. In the questionnaire, participants were shown actual examples of the different types of visualisations before being asked about their preferences. The results revealed that 69% of participants thought (giving a rating of 4 or 5 out of 5). that the textual representation of rules (e.g. "I turned on the fan because the temperature is hot and the humidity is high") were a clear and easy way of visualising the rules.

Alternatively, 59% of participants felt that the decision tree was a clear representation and 62% of participants thought that the use of colour (as illustrated in the questionnaire) helped with the comprehensibility of the decision tree.

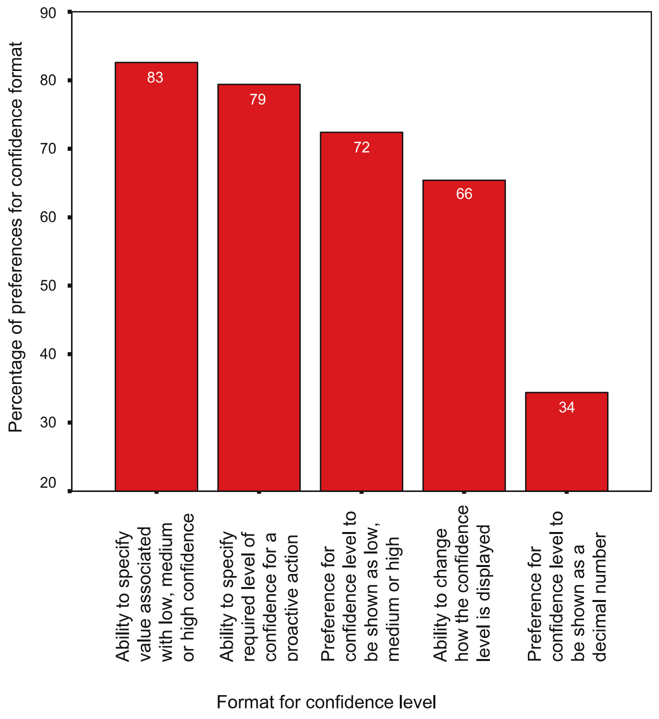


Figure 16. Users' strong preferences with respect to various forms of visualising the confidence level associated with rules governing the system's proactive behaviour.

Only 10% of users felt that the IF-THEN rule table approach (illustrated in the questionnaire using a table similar to that shown in Figure 11) was a clear way of representing the rules.

It was interesting to find that more than 60% of participants held a strong preference for having the option of viewing their context history "occasionally and only when I request it".

6.2.3.3. *Confidence Level Visualisation.* Another way to make a system's decisions transparent to its users is by making the confidence level of a suggestion visible (e.g. Figure 8). More than 50% of study participants agreed that the confidence level offers a basis for understanding the system and the accuracy of its judgements. For this questionnaire study, we considered two modalities of visualising the confidence level: decimal format and categorical format, i.e. 'high', 'medium' or 'low'. In Figure 16 the bars represent the percentage of users who strongly favour different forms for visualising the confidence level of rules embedded in the user model (giving a rating of 4 or 5 out of 5).

6.2.4. *Exploring Attitudes towards Deployment in Home vs. Office Environments*

Participants appeared to be more comfortable having an intelligent control system at home rather than at work. More participants considered that the system control should be diminished in a work environment compared to a home environment. This attitude may be fostered by the users' need to be more in control in a work environment. For example, more than 60% would like to have an automatic light control in their rooms and only 33% in their office. This attitude is also reflected in the users' attitude regarding the perceived market value for such a system.

6.2.5. *Exploring the Attitudes of Expert Users*

In our study, we found that the subjects who were more experienced in terms of computer knowledge were significantly more interested in specifying a confidence level threshold for proactive action ($t(22) = 2.20$, $p < 0.05$) and in controlling the system when necessary ($t(28) = 2.83$, $p < 0.01$). The reduced number of degrees of freedom for the first case is due to missing data.

The need to control the system correlates with the amount of knowledge users hold about computers ($r(28) = 0.48$, $p < 0.01$): the higher their level of expertise, the more in control they would like to be.

6.2.6. *Qualitative Analysis*

The questionnaire design enabled participants to provide comments when answering each of the Likert scale questions. These comments are summarised below.

When commenting about whether the proactive system should prompt before carrying out an action or whether the system should simply carry out the action automatically, participants commented that this should be task-dependent, given the possible negative consequences related to the automatic actions, e.g. "fan blowing papers".

The option of having the system prompting for a short time and then not performing the action is usually considered as having little value, particularly "if the certainty level is high, it makes sense to let the system get on with it".

One interesting comment made by a participant was as follows: "I don't think that the user of a washing machine is interested in knowing how the system works out when to turn the water supply off, but knowing that there is a facility to cope with half loads, i.e. use less water, is useful information". This comment underlines the dichotomy between two types of user models: the structural model that assumes that the user has internalised the structure of how the system works, and the functional model that assumes that the user has internalised procedural knowledge about how a system can be used. Users holding a functional model of an intelligent control system are probably more inclined to accept its learning curve and have a high tolerance to its errors. In fact, study findings suggest that participants previously exposed to proactive systems are significantly more willing to accept the system making wrong decisions ($t(28) = 1.97$, $p < 0.05$).

We believe that these findings indicate that the designers of systems such as the IOS should anticipate a significant variance in user preferences for different system features, particularly scrutability and control. One way to successfully accommodate such a large range of user preferences consists of designing a system able to tailor itself to each individual user. However, in practice, this may be difficult to implement, given the richness and uniqueness characterising each individual user. A possible way to approach this problem, open to future work, consists of identifying groups of users which differ greatly in terms of their preferences for system behaviour, i.e. scrutability and enabled control. Following such an approach could lead designers to develop a range of different systems, each designed on the basis of the requirements of a distinctive category of user.

7. Discussion and Lessons Learned

We would argue that our current work on the IOS represents one of the first case studies of a truly proactive system: a system that combines context-awareness, machine learning and support for scrutability. The following subsections describe the issues explored and lessons learned from our research to date with the IOS.

7.1. BENEFITS AND DRAWBACKS OF DEPLOYMENT

From our experience in developing and deploying ubicomp/context-aware systems we are of the strong opinion that real (albeit prototype) deployment is crucial to developing insights and highlighting areas for further investigation which can all too easily be missed as part of a purely paper study. However, the difficulty of developing a prototype system that can run continuously 24/7 for many weeks, if not months, should not be underestimated. The current system has undergone numerous modifications in order to reach a level where the user interface and overall system reliability are at an acceptable level for deployment with nontechnical users. Unforeseen problems can, of course, also occur, For example, the initial system had been developed to turn on the fan but not to turn on a heater. However, a move to a new building brought cooler conditions causing a slight system modification to control heaters also. However, on the positive side, deployment did enable users to make explicit and meaningful choices. The layout of a user's office has a significant impact on how she may alter her preferences for the system's behaviour. For example, one of the authors has a preference for the IOS to turn her office heater on without prompting, but she does request prompting before the system turns on the fan (because she may have papers in front of the fan). However, another user of the system has these preferences reversed.

Furthermore, it has only been through actual deployment that we have come to realise the importance of practical issues such as the need to deploy prototype systems that do not increase office clutter, or additional computer noise. It was also through deployment that we realised the strong preference (amongst the technical users involved in the study) for a system that supported touch screen based interaction.

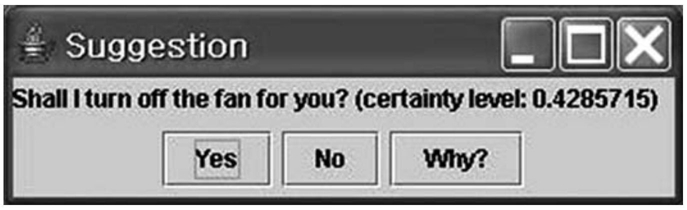
7.2. ACKNOWLEDGING THE NEED FOR STRONG USER BENEFIT TO ENSURE USER ADOPTION

Following on from the arguments made in the previous section, we have also found it crucial when asking users to deploy prototype systems in or around their offices that they receive a genuine benefit from adopting the given technology. For example, we experienced this as a key issue when asking work colleagues to cooperate in the deployment of a collection of electronic office door displays (Cheverst et al., 2003). With research such as this, it is critical that the cost of using the system is not too high to outweigh perceived benefits in the short term. The learning approach employed by the IOS relies on the capture of events such as the user turning on a fan, turning off a heater etc. It is simply not realistic to expect a user, even one with a vested interest in the experiment, to go through the trouble of performing two tasks in order to turn on a device, i.e. Task 1: turning on the fan and then Task 2: pressing a button on a GUI in order to let the system record the fact that she has just turned on the fan. This was the case with an earlier version of the IOS; in order to reduce this cost we first tried to sense automatically (using the sensor described in Section 4.1.3) when a device was turned on (thus removing Task 2). However, this approach alone has some difficulties, because once the fan was turned off manually, the system would not be able to turn on the fan (using X10). To counter this problem, we developed a GUI for the system that enabled the user to turn on the device using a GUI on a separate touch-sensitive display (i.e., the one shown in Figure 7).

7.3. COMPLEXITY VS. COMPREHENSIBILITY

There is an inherent conflict between the comprehensibility of a system and the complexity of the algorithms used to generate the desired behaviour. A simple approach may be easy to comprehend but may not produce the desired behaviour. Conversely, a more complex approach achieving accurate results may not be easily understood by the average user.

From our experience with the IOS, we feel that the use of fuzzy decision trees offer a good compromise between complexity and comprehensibility. While the full details of the low-level computations may be beyond an average user, the resulting fuzzy decision tree is reasonably easy to understand and most importantly enables the user to scrutinise the underlying rules that the system has learnt and which it will use to perform proactive activation of devices. Furthermore, unlike a classical decision tree based on crisp cut-points, the fuzzy variant can generate (a ranked list of) secondary reasons for a decision. In this way, it not only provides more detail about why a particular decision was taken but it also does so using a human conceptualisation, i.e. fuzzy categories. For example one can imagine thinking: “yeah it is a bit hot but still somewhat mild”.



Shall I turn off the fan for you? (certainty level: 0.4285715)

Figure 17. An early version of a (much disliked) modal dialog box for prompting the user.

7.4. INTERRUPTION AND USER CONTROL

Uncontrollability and unpredictability are key factors responsible for the negative and stressful impact caused by interruptions (Cohen, 1980). Our approach is to reduce these negative effects by providing a user of the system with appropriate levels of control and comprehensibility (thus helping her to better predict system behaviour).

The IOS enables the user to choose whether to be informed before a system proactive behaviour takes place. For example, Figure 8 shows the prompt displayed by the system when the user has requested to be informed before her fan is turned on. In effect these prompts can be viewed as interruptions to the user. Users can control (over time) the extent to which these prompts can occur. For example, in the current version of our system, as a user learns to increase her trust in the system's decisions, she may choose (using the check boxes shown in Figures 8) less prompting by the system. By doing so, she is effectively negotiating the level of interruption occurring (McFarlane and Latorella, 2002) and also sacrificing greater control in order to permit more automatic behaviour (through the actuation of devices) to occur.

The IOS also provides the user with information regarding its learnt behaviour. For example, Figure 8 shows the information (specifically the message "I believe that you want to turn the fan off (confidence level: High)" presented to the user when a rule to turn off the fan is triggered. Again, the provision of such information can be viewed as a form of interruption. But the user may chose to ignore this information or to act on it by, in this case, pressing the 'OFF' button.

An earlier version of the system used a significantly more disruptive approach to prompting the user: a modal dialog box such as that shown in Figure 17, which was designed to appear on the user's primary desktop monitor. The change to a less disruptive interface was made following user feedback from those running this earlier version of the system.

In addition to providing prompts regarding the system's learnt behaviour, the IOS also attempts to promote comprehensibility (and thus reduce its unpredictability) by supporting scrutability, i.e. enabling the user to access the rules governing system behaviour. For example, Figure 8 illustrates how the user is given the opportunity to scrutinise the currently triggered behaviour.

8. Future Work

Our immediate future work is to deploy the system (or at least a variation of it) into the offices of less technical users and also to move towards deployment in a home environment for extended periods of use/study. It will be interesting to observe whether there are significant differences to user acceptance of the system in home and office environments, given the findings of the questionnaire-based study. We hope that the availability of the software part of the IOS on the web will also enable other groups to extend the system and perform their own deployments and experimentation.

Furthermore, we want to extend the number of attributes captured by the system. This will not only enable the system to reason on the basis of richer information about the current state of the environment but it should also prove beneficial in dealing with values fluctuating around a cut-point (as decisions will not be based on a single attribute alone). We are also interested in incorporating time-related factors such as the time of day, the day of the week and perhaps even the time of year. These attributes will help to capture more accurately the patterns of user behaviour, such as daily or seasonal routines.

9. Summary and Concluding Remarks

This article has described our exploration into the feasibility of utilising context history with a fuzzy-decision-tree-based machine learning algorithm in order to support personalised but comprehensible proactive behaviour in an office environment.

One key motivation for carrying out this research has been to explore both the technical and human factors issues associated with a context-aware proactive system such as the IOS. Consequently, a core element of our research has been to explore issues of user control, comprehensibility and interruption. Comprehensibility in the IOS is supported by allowing the user to scrutinise the rules learnt by the system and the context history used to generate the rules (if this level of detail is requested by the user). Our questionnaire study revealed that in general participants were keen to scrutinise behaviour in this way. Furthermore, users involved in the study had a positive response to the concepts behind the IOS but clearly expressed a desire for maintaining levels of control over the system's proactive behaviour.

By designing and deploying the Intelligent Office System we have achieved a validation of the feasibility of collecting context history (using both 'off the shelf' and tailored sensing devices) and utilising this context history in order to learn patterns of user behaviour which can, in turn, be used to drive proactive behaviour (e.g. turning on a fan in the user's office).

We strongly believe that through the actual deployment of a working system we have managed to obtain far greater confidence in our understanding of the technical and human factors issues than would have been achievable had only partial system implementation and Wizard of Oz based studies taken place.

Acknowledgements

We would like to thank the reviewers of this paper and the guest editors of this special issue for their highly constructive comments. We would also wish to acknowledge the help provided by Martyn Burgess from Lancaster University. This research has, in part, been part funded by the EPSRC CASIDE project.

References

1. Abowd, G.D. and Mynatt, E.D.: 2000, Charting past, present and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction, Special Issue on HCI in the New Millenium* **7**(1), 29–58.
2. Arroyo, E., Selker, T. and Stouffs, A.: 2002, Interruptions as multimodal outputs: Which are the less disruptive? *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces (ICMI'02)*. Pittsburgh, Pennsylvania, pp. 479–482.
3. Barkhuus, L. and Dey, A.K.: 2003, Is context-aware computing taking control away from the user? Three levels of interactivity examined. *Proceedings of UbiComp 2003: Ubiquitous Computing*. Seattle, USA: Springer-Verlag, pp. 159–166.
4. Brumitt, B., Meyers, B., Krumm, J., Kern, A. and Shafer, S.: 2000, EasyLiving: technologies for intelligent environments. *Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing (HUC 2000)*. Bristol, UK: Springer-Verlag, pp. 12–29.
5. Byun, H.E. and Cheverst, K.: 2003, Supporting proactive "Intelligent" behaviour: the problem of uncertainty. *Proceedings of the UM03 Workshop on User Modeling for Ubiquitous Computing*. Johnstown, PA, pp. 17–25.
6. Byun, H.E. and Cheverst, K.: 2004, Utilising context history to provide dynamic adaptations. *Journal of Applied Artificial Intelligence* **18**(6), 533–548.
7. Cheverst, K., Davies, N., Mitchell, K. and Efstratiou, C.: 2001, Using context as a crystal ball: rewards and pitfalls. *Personal Technologies* **3**(5), 8–11.
8. Cheverst, K., Dix, A., Fitton, D. and Rouncefield, M.: 2003, "Out To Lunch": exploring the sharing of personal context through office door displays. *Proceedings of the International Conference of the Australian Computer-Human Interaction Special Interest Group (OzCHI'03)*. Brisbane, Australia: IEEE Computer Society Press, pp. 74–83.
9. Coen, M.: 1998, Design principles for intelligent environments. *Proceedings of the Tenth Conference on Innovative Applications of Artificial Intelligence*. Madison, WI: AAAI Press, pp. 37–43.
10. Cohen, S.: 1980, After-effects of stress on human performance and social behavior: A review of research and theory. *Psychological Bulletin* **88**, 82–108.
11. Dey, A.K. and Abowd, G.D.: 2000, The context toolkit: aiding the development of context-enabled applications. *Proceedings of the Workshop on Software Engineering for Wearable and Pervasive Computing*. Limerick, Ireland.
12. Guetova, M., Holldobler, S. and Storr, H.: 2002, Incremental fuzzy decision trees. *Proceedings of the 25th German Conference on Artificial Intelligence (KI2002)*. Aachen, Germany, pp. 67–81.

13. Holmquist, L.E., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M. and Gellersen, H.W.: 2001, Smart-its friends: a technique for users to easily establish connections between smart artefacts. *Proceedings of UbiComp 2001: Ubiquitous Computing*. Atlanta, USA, pp. 116–122.
14. Horvitz, E.: 1999, Principles of mixed-initiative user interfaces. *Proceedings of the CHI 1999 Conference on Human Factors in Computing Systems*. New York: ACM Press, pp. 159–166.
15. Intille, S.S.: 2002, Designing a home of the future. *IEEE Pervasive Computing* April–June, 80–86.
16. Intille, S.S. and Larson, K.: 2003, Designing and evaluating supportive technology for homes. *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. Kobe, Japan: IEEE Press.
17. Jameson, A., Baldes, S., Bauer, M. and Kroner, A.: 2004, Resolving the tension between invisibility and transparency. *Proceedings of 1st International Workshop on Invisible and Transparent Interfaces*. Gallipoli, Italy, pp. 29–33.
18. Janikow, C.Z.: 1996, Exemplar learning in fuzzy decision trees. *Proceedings of the Conference on Fuzzy Systems (FUZZIEEE '96)*. New Orleans, pp. 1500–1505.
19. Karlgren, J., Hook, K., Lanz, A., Palme, J. and Pargman, D.: 1994, The glass box user model for information filtering. *Proceedings of the 4th International Conference on User Modeling (UM'94)*. Hyannis, MA.
20. Kay, J., Kummerfeld, R.J., and Lauder, P.: 2003, Managing private user models and shared personas. *Proceedings of the UM03 Workshop on User Modeling for Ubiquitous Computing*. Johnstown, PA, pp. 1–11.
21. Mantyjarvi, J. and Seppanen, T.: 2002, Adapting applications in mobile terminals using fuzzy context information. *Fourth International Symposium on Human-Computer Interaction with Mobile Devices (MobileHCI 2002)*. Pisa, Italy: Springer-Verlag, pp. 95–107.
22. McFarlane D.C. and Latorella, K.A.: 2002, The scope and importance of human interruption in human-computer interaction design. *Human-Computer Interaction* **17**(1), 1–61.
23. Mitchell, T.M., Caruana, R., Freitag, D., McDermott, J. and Zabowski, D.: 1994, Experience with a learning personal assistant. *Communications of the ACM* **37**(7), 80–91.
24. Mozer, M.C. and Miller, D.: 1998, Parsing the stream of time: the value of event-based segmentation in a complex real-world control problem. In: C. L. Giles and M. Gori (eds.), *Adaptive Processing of Temporal Information*. Berlin: Springer, pp. 370–388.
25. Pohl. W.: 1996, Learning about the user – user modeling and machine learning. *Proceedings of Machine Learning Meets Human-Computer Interaction (ICML'96 Workshop)*. pp. 29–40.
26. Rodin, J. and Langer, E.: 1977, Long-term effects of a control-relevant intervention with the institutionalized aged. *Journal of Personality and Social Psychology* **35**(12), 897–902.
27. Rubinstein, J.S., Meyer, D.E. and Evans, J.E.: 2001, Executive control of cognitive processes in task switching. *Journal of Experimental Psychology: Human Perception and Performance* **27**(4), 763–796.
28. Salovaara, A. and Oulasvirta, A.: 2004, Six modes of proactive resource management: a user-centric typology for proactive behaviors. *Proceedings of the Third Nordic conference on Human-Computer Interaction*. Tampere, Finland: ACM Press, pp. 57–60.

29. Shiu S.C.K., Sun, C.H., Wang, X.Z. and Yeung, D.S.: 2000, Maintaining case-based reasoning system using fuzzy decision trees. *Advances in Case-Based Reasoning, 5th European Workshop (EWCBR 2000)*. Trento, Italy: Springer, pp. 285–296.
30. Speier, C., Valacich, J.S. and Vessey, I.: 1997, The effects of task interruption and information presentation on individual decision making. *Proceedings of the 18th International Conference on Information Systems*. New York: Association for Computing Machinery, pp. 21–36.
31. Weiser, M.: 1991, The computer for the 21st century. *Scientific American* **265**(3), 94–104.
32. X10 Limited: 2004, What is X10? http://www.smarthome.com/about_x10.html
33. Zeidler, J. and Schlosser, M.: 1996, Continuous-valued attributes in fuzzy decision trees. *Proceedings of the 6th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems.* Granada, Spain, pp. 395–400.

Authors' Vitae

Dr. K. Cheverst
Department of Computing, Lancaster University, Lancaster, UK. Keith Cheverst is a Senior Lecturer with the Computing Department of Lancaster University. His research interests span: Mobile and Ubiquitous Computing technologies, Human Computer Interaction and CSCW and he has published over 80 research papers across these fields. He is also a keen advocate of User Centered Design and is currently the Principal Investigator on the EPSRC funded CASIDE project (www.caside.lancs.ac.uk).exploring the design and use of situated display technologies.

Mr. Hee Eon Byun
Department of Computing, Lancaster University, Lancaster, UK. Hee Eon Byun is a Ph.D. candidate in Computing at Lancaster University. He received his BSc. in Computer Science from Sogang University in 1989, and his M.Sc. degree in Environmental and Ecological Sciences from Lancaster University in 1999. His primary interests lie in the areas of user modelling and machine learning. The research for his thesis work focuses on the utilisation of context history and machine learning in order to support proactive services and the HCI implications associated with such an approach.

Dr. Corina Sas
Department of Computing, Lancaster University, Lancaster, UK. Corina Sas is a lecturer in HCI at the Computing Department, Lancaster University. She received her bachelor degrees in Computer Science and Psychology and masters degree in Industrial Psychology from Romania. She received her PhD in Computer Science from University College Dublin in 2004. Her research interests include user modelling, adaptive systems and usability studies. She has published in journals and international conferences in these areas.

Mr. Daniel Fitton
Department of Computing, Lancaster University, Lancaster, UK. Daniel Fitton is currently a research associate on the EPSRC funded CASIDE project (which aims to investigate such issues as how situated display technology can influences and facilitates coordination and community). His research mainly involves the rapid-prototyping, deployment and investigation into the use of new and novel situated display based systems for use in the ′real world′. He is presently in the process of writing up his thesis part-time, which focuses on the support of asynchronous messaging using interactive situated technology.

Dr. Christian Kray
Department of Computing, Lancaster University, Lancaster, UK. Christian Kray received his diploma degree in Computer Science from Saarland University in 1998, and his Ph.D. in Computer Science from Saarland University in 2003. He has worked as a full-time researcher at the German Research Centre for Artificial Intelligence (DFKI) in Saarbruecken, Germany, where his research focussed on situated interaction on spatial topics in the context of mobile applications. Since 2003 he has beena post-doctoral researcher at Lancaster University, where he investigates spatial reasoning and novel interaction techniques in the context of ubiquitous computing. His contribution is based on his background in applied AI and experiences gained from deploying a number of prototypical applications with novel interaction metaphors.

Mr. Nicolas Villar
Department of Computing, Lancaster University, Lancaster, UK. Nicolas Villar is a research associate and part-time PhD student in the Embedded Interactive Systems research group of the Lancaster University Computing Department. His interests lie in developing the technologies to enable novel forms of human computer interaction. In particular, his research explores the possibilities of tangible interfaces and other forms of interaction through real world objects.