Harnessing Context to Support Proactive Behaviours

Hee Eon Byun and Keith Cheverst

Distributed Multimedia Research Group,
Department of Computing,
Lancaster University
Lancaster, LA1 4YR
e-mail: {byunh, kc}@comp.lancs.ac.uk

ABSTRACT

In order to realise an intimate and proactive personal assistant under the Weiser's vision for ubiquitous computing environments, we propose the utilisation of context history together with user modelling and machine learning techniques. Our approach could support dynamic adaptations to changes of the user's life style or changes in the situation itself by detecting patterns in a user's behaviour. In addition, we propose the requirement of an explicit explanation about such dynamic adaptations to the user in order to encourage a trust relationship between the user and the context-aware system. In this paper, we describe appropriate scenarios that reveal the potential of our approach. In order to examine how dynamic adaptations can be implemented, three types of learning design are proposed and the implications of using these designs are discussed.

1. Introduction

Since Weiser [3] first suggested the concept of ubiquitous computing, context-aware computing has become one approach for increasing the productivity or welfare of users situated in ubiquitous computing environments. The value of using context has been demonstrated in a number of context-aware applications, which mainly focus on providing generally "*reactive*" adaptations to the user's personal and environmental context. However, one can argue that the vision that Weiser had for ubiquitous computing environments, i.e. a vision of "intimate computing" [4] supporting "human assistance" [5] has yet to arrive. Indeed "intelligent" applications to support the kind of scenario described in [6] involving Sal (e.g. "*proactively*" finding and then reserving Sal a parking place at work) are still awaiting an appearance.

The term 'Context-aware' has been defined as "systems [that] adapt according to the location of user, the collection of nearby people, hosts, and accessible devices, as well as to changes to such things over time" [1]. Besides this definition, a number of definitions can be found in the literature. However, the main concept of context-awareness is the adaptation of systems to the user or his/her environment by capturing and understanding contexts, which can be, for example, the user's preferences, location, temperature of the environment, network connectivity of the user's device, etc. In a broad definition, everything that can describe the characteristics of the user and the situations of his/her environments can be considered as context.

Currently, the usage of context has been largely confined within the following two categories: (i) "using raw context" and (ii) "interpretation of context using a group of context". The clearest example of (i) is illustrated by most location-aware applications. In such applications, location is used as it is, although the representation of location can be changed, for example, from symbolic to coordinate or vice versa. As an example of (ii), the Context Toolkit [1] provides a generalised mechanism for the interpretation of groups of context. In more detail, context aggregators gather all necessary contexts and then context abstractors interpret the group of context using predefined rules. For example, if the sound level in a meeting room is high and there are some people in the room, it can be assumed that a meeting is being held in the room [1].

Another limitation of the current generation of context-aware applications is that most applications are only concerned with the present context (only a few exceptions exist that relate to some form of context-based information retrieval) [2]. However, the history of contexts might be extremely valuable and could enable us to go beyond the current level of context interpretation. Such context history could provide far more information about the user. In more detail, by noticing patterns from the context history (including the user's behaviour) a system could exhibit "intelligent" or more specifically "proactive" behaviour. For example, having determined that the user has a regular meeting schedule, a system could remind the user of that event. The utilisation of context data history in this way appears to be a relatively under-explored research field.

In this paper, "intelligent" behaviour will be defined as acting proactively on induced knowledge. For example, if an application can induce the life patterns of the occupants of a house, it can optimise energy costs of the house. This "intelligent" behaviour will largely depend on knowledge on the user's habits and goals [7], [8].

In order to extend the current state of context-aware computing to one that is capable of supporting the development of "intelligent" applications, our approach has been to exploit the research areas of user modelling and machine learning (see figure 1). The area of user modelling has been used to enable the adaptation and personalisation of services or information delivered to users, e.g., adaptive hypermedia [9]. The main concept of context-awareness is the adaptation of systems to the user or his/her environments by capturing and understanding contexts. Thus, it might be said that context-aware computing and user modelling have a common goal: that is tailored adaptation. But yet the application areas of each topic currently seem largely unrelated.

The main concern of machine learning is to learn some knowledge or rules from past experiences. Knowledge or rules learned might be a user's individual characteristics. For example, the CAP (Calendar APprentice) application learns a user's meeting preferences e.g., preferences duration, time, location, etc.[8] and uses this information to assist the user in the process of scheduling meetings. Thus, machine learning has been considered as a practical method for the representation and acquisition method for user models. 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, etc. in order to adapt the services to the user's individual characteristics (examples include Mitchell et al [7], Pohl [10], Billsus and Pazzani [11], and Ruvini and Fagot [12]).

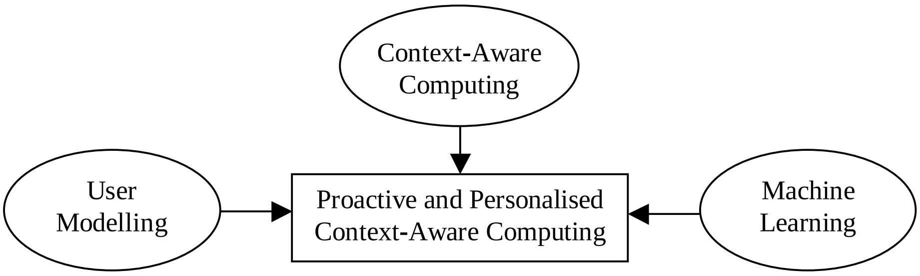

Figure 1. A synergistic combination of three distinct research areas.

Anhalt et al [13] and Selker and Burleson [14] also suggest the proactive behaviour of their context-aware applications. However, in these applications, the proactive behaviours mainly depend on predefined rules, which are not suitable for providing dynamic adaptations in an environment where the predefined rules should be adjusted according to changes of the user's life style or changes in the situation itself. We believe that our chosen approach, i.e. utilising context history together with user modelling and machine learning techniques can be used to infer patterns from the user's behaviour.

One implication of our proposed approach is that the user might not be able to understand the proactive behaviour of context-aware devices or applications. Consequently, the provision of an explicit and understandable explanation for a proactive behaviour to the user is desirable. Furthermore, providing the user with an explanation of the system's proactive behaviour can enable the user to provide some level of feedback that may in turn be used by the machine learning algorithms employed. It is important to note that the provision of explanations to the user raises an important requirement on the type of learning methods that may be used. In more detail, learning based on decision tree learning is suitable because the rules generated may be intelligible to humans. However, learning based on the neural network approach is less suitable because the weights produced by this approach are difficult to interpret by human users [7].

The structure of the remainder of this paper is as follows. In section 2, we derive appropriate scenarios that can reveal the potential of utilising user modelling and machine learning techniques within context-aware applications. In section 3, three types of learning design are proposed and the experiment with one of the designs is explained. Subsequently, several implications discovered by the experiment and future works are provided in section 4. In section 5, related works are investigated and analysed. Finally, our conclusions at this stage of the research are presented.

2. Scenarios

2.1 Intelligent Personal Services

The starting scenario for this research was the Personal Digital Secretary (PDS) [15]. This application idea emerged from considering how the classical remembrance agents, for example, Forget-me-not [16] and CyberMinder [17], might be extended using the techniques of user modelling and machine learning in order to support a user's daily activities in an everyday computing setting [18]. The PDS was designed based on the assumption that it would support user's daily activities beyond the role of a remembrance agent or reminder. The conceptual structure of our PDS is composed of five main modules as shown in Figure 2. For the detailed explanations for each module, please refer to [15].

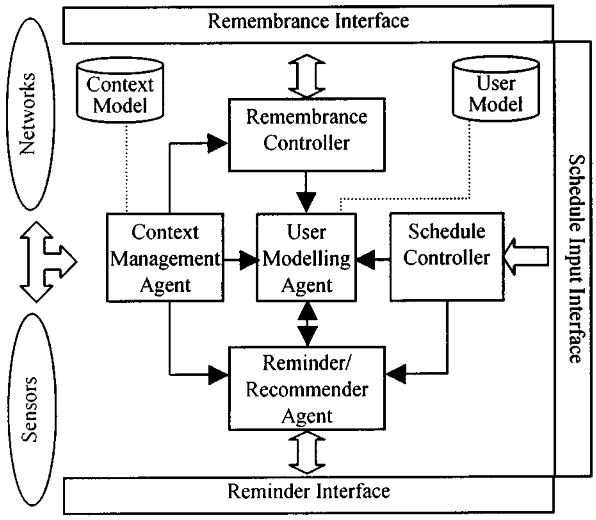

Figure 2. Conceptual Structure of The Personal Digital Secretary.

The scenario proved extremely useful for helping us to explore how the paradigm of context-aware computing could be usefully augmented by the utilisation of user modelling and machine learning techniques. Some of the scenarios that can reveal proactive usages of the PDS are described below.

- *Scenario A:* When a user passes by a theatre, the PDS can notify the user that the theatre is playing one of the user's favourite movies. This type of functionality could be realised by a context-aware system (such as GUIDE [19]) by utilising both the location context and information about the user's preference.
- *Scenario B:* If a user is in an intelligent environment, for example, in an intelligent home [20] and the user commands some action, for example, 'close curtains in the living room', the PDS could modify its user model and deliver this command to the appropriate agent in the home. As a result, the system could, over time, learn that an appropriate context-aware behaviour is to close the curtains when it gets dark outside.
- *Scenario C:* If a user participates in a meeting at 10 am every forth Monday, the PDS might learn the pattern of this regular meeting and remind the user to prepare for the meeting at an appropriate time. However, a more sophisticated level of learning would be desirable in order to enable the system to realise when such a notification is inappropriate, for example, when the user is on holiday.
- *Scenario D*: if a user makes a rule to hold the room key when leaving his/her office after 6pm, this can be captured in a user model. Consequently, when the user is about to leave his/her office without the room key after 6pm, the PDS could warn the user before he/she gets locked out of the office!.

Through the conceptual design and scenarios of the PDS, the proactive behaviours can be broadly categorised into two types: "*proactive modelling-based adaptation*" and "*proactive rule-based adaptation*". The first type of proactive behaviour provides adaptations based on the current context and the patterns/rules inferred from the history of the user's behaviour and the second means providing adaptations based on the current context and predefined rules.

2.2 Calculating the Level of Security Risk in a User's Office

In order to examine how the two types of proactive behaviour can be designed and implemented, we derived a very specific scenario.

> "The user is an academic staff member in a university. The staff members use MediaCup devices [21] that can sense the temperature of its containing liquid and the status of how it is being used (e.g., placed on a surface or carried). When the door of the user's office is open and the user has left the office, this situation is generally considered as a case of high security risk. If the security level is 'high', a security-warning message will be delivered to the user."

However, if the MediaCup has hot coffee and is located in the office, this situation might suggest that the user will return to his/her office within a short time. Therefore, the final decision on the level of security risk of the user's office can be influenced by the situation. Such considerations for the exceptional cases (a cup of hot coffee in the office) against the general rules (the cases of high security risk) may be necessary in order to reduce frustrating intrusions (inappropriately raising security warnings) into the user's life. These exceptional cases can be expressed as a set of rules or can be induced from the past history of the user's behaviour.

Such past history is not exploited by current approaches for achieving context interpretation. For example, context aggregation [1], context synthesis [22] and context fusion [23], [24] only utilise current contexts, e.g. the current location or current time of day. In order for current approaches to consider such exceptional cases, a number of predefined rules must be adopted. However, such predefined rules may frequently need to change in order to reflect changes to exceptional cases. Our approach, that of utilising the past history of a user's behaviour is presented in the next section.

3. Approach

In this section, we will derive three kinds of inference design in order to obtain a higher level conceptual context, i.e., the security level of the user's office. The designs will be described in turn.

3.1 Use of a Single Learning Algorithm

This design uses a single learning algorithm as depicted in figure 3 and therefore inferring the conceptual context from this design is very straightforward. When a new situation arises, the single machine learning algorithm induces the security level of the new situation based on the context history. Then, the new situation and the security level inferred are stored in the context history as a recent instance. According to the result from learning, a security-warning message could be delivered to the user.

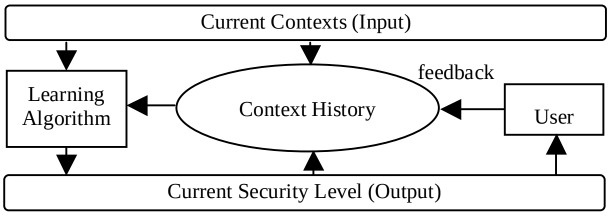

Figure 3. The use of a single learning algorithm.

The actual situation to be inferred might be more complicated than the scenarios that can be solved by a one-size-fit-all method. For example, even though the user was not in his/her office and the door open, it might be the case that the user was talking with somebody in the corridor for a moment. In this case, the user could give feedback to the system, in order to let the system know that "I am having a small meeting with colleagues in front of my office therefore do not raise a security-warning message in this situation in the future".

3.2 Use of Predefined Rules and a Single Learning Algorithm

In order to refine the behaviour of our learning system, a set of rules for the decision on the security level can be predefined and a machine learning algorithm can be used to figure out more specific exceptional cases against the rules as shown in figure 4. In this design, the first decision on the security level is made by the predefined rules. However, according to the result from the learning algorithm whose purpose is to consider the exceptional cases, the final decision can be changed.

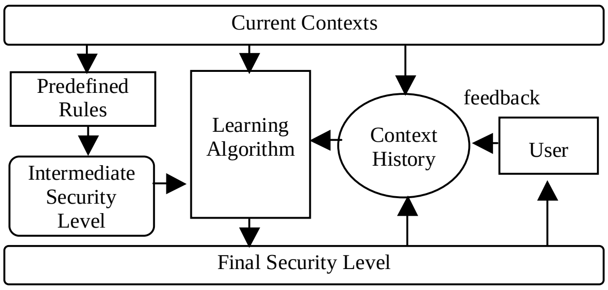

Figure 4. The use of predefined rules and a single learning algorithm.

We conducted an experiment to examine this design. For our experiment, it was first necessary for us to define a set of rules for the general security level of the office. The rules were defined as follows:

- *High security risk*: The door is open when the user has left the office during his/her office hours.
- *Low security risk*: The door is closed (but not locked) when the user has left the office during his/her office hours.

Secondly, we needed to develop a system that could ascertain the probability of the user returning to the office within a short period of time. The steps taken for designing a suitable learning system are described below.

- *Step 1* is to define a target function that is the type of knowledge to be learned. In this case, the target function is defined as "Will the user's location be 'the office' in the near future?"
- *Step2* is to decide a hypothesis space (possible value range of the target function). The hypotheses for the target function are defined as "v_1 = true" and "v_2 = false".

- *Step3* is to establish an appropriate training data set (context history). A training data set is composed of a set of instances (records), which contain a set of attributes (contexts) relevant to the target function. Although there could be a number of attributes that are likely to be relevant to our target function, in order to focus on the relations between the user's location and the status of the MediaCup, we considered just two attributes i.e., "*UserInOffice*" and "*TempCup*". The context history based on these two attributes is presented in table 1.

Time Stamp	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
UserInOffice	true	true	true	false	true	true	false	false	false	true	true	true
TempCup	cold	hot	hot	hot	hot	cold	cold	cold	cold	cold	hot	hot

Table 1. Context history as a training data set.

- *Step 4* is to select or design an appropriate learning algorithm. We selected the Naïve Bayes Classifier in order to obtain explicit probabilities for each hypothesis.

For the next time period of this context history, i.e., "t13", if the user is not in the office and the door open, the intermediate security level becomes "high" based on the predefined rules. Subsequently, this high level of security risk will trigger a test for exceptional cases. If the new situation raised at this time indicates *UserInOffice* = false and *TempCup* = hot, the Naïve Bayes algorithm will calculate the probabilities $P(v_j)$, $P(a_i|v_j)$, and finally v_{NB} = argmax $P(v_j)\Pi P(a_i|v_j)$ [8] as follows:

- $P(v_j)$: the probabilities of the different target values, based on their frequencies over the training data set
  P(UserInOffice = true) = 8/12 = 0.667
  P(UserInOffice = false) = 4/12 = 0.333
- $P(a_i|v_j)$: the conditional probabilities for each combination of attributes and target values, based on their frequencies over the training data set
  P(TempCup = hot | UserInOffice = true) = 5/12 = 0.417
  P(TempCup = hot | UserInOffice = false) = 1/12 = 0.083
- The probabilities for each hypothesis
  v_1 = P(UserInOffice = true) × P(TempCup = hot | UserInOffice = true) = 0.278
  v_2 = P(UserInOffice = false) × P(TempCup = hot | UserInOffice = false) = 0.028

In this situation, the Naïve Bayes Classifier calculates the conditional probability of the user being in his/her office (0.278/(0.278+0.028) = 0.908) as higher than that of the user being in another place (0.028/(0.278+0.028) = 0.092). Based on this result, it could be reasoned that for the time period "t14" the user is likely to return to the office given that the MediaCup contains hot coffee. Therefore, if we can accept this probabilistic decision, the final security level could be changed from "high" to "low", hence no security warning message would be raised even though the user has left the office and left the door open. If the user chooses to query this behaviour, the system could provide an explanation such as: "Because your MediaCup contains hot coffee and is located in your office, the security level was set to 'low'."

3.3 Use of Multi Learning Algorithms

With this design, the interpretation process is composed of several learning algorithms as depicted in figure 5. In general, the selection of learning algorithms depends on the characteristics of the situations to be learned. For example, the Bayesian method can be adopted when multiple hypotheses are needed (not just for "yes" or "no") and a set of rules can be extracted from a decision tree. Therefore the decision tree learning method is suitable for learning algorithm 1 and the Naïve Bayes Classifier is suitable for learning algorithm 2.

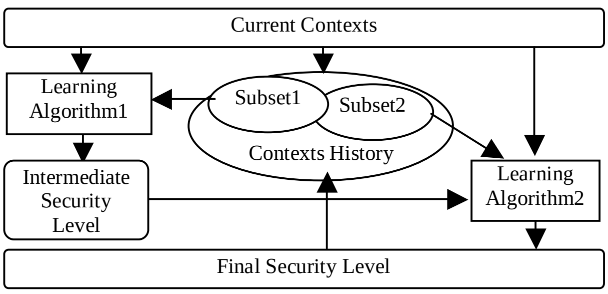

Figure 5. The use of multi learning algorithms.

4. Findings and Future Work

The basic assumption of probabilistic learning methods (e.g., Naïve Bayes Classifier) is that probability distributions are the basis of inference and, therefore, reasoning about these probabilities for a new situation can lead to an optimal decision for the new situation [8]. In addition, inductive learning methods (e.g., decision tree learning) are based on the generalisation of patterns. In more detail, identifying features (e.g., the user is in the office) are generalised from some observed training data set (e.g., our context history). Consequently, although the probabilistic result from the Naïve Bayes Classifier or the rules from the decision tree learning indicate that the user is likely to be in the office, we clearly cannot state (with absolute certainty) that the user will return to the office for the next time period. The decision based on these learning methods intrinsically holds uncertainty.

Another implication is that the experiment described in the previous section seems to be over simplified. In order to extract patterns of the user's behaviour more precisely, temporal elements (the transition period of the changes in the status of context) ought to be considered as another dimension. In more detail, the coincidence of transitions for the value of *UserInOffice* (from false to true) and for the value of *TempCup* (from hot to cold) can allow us to measure more precisely the likelihood of the user returning to the office in the near future. For example, given that the transition period of *UserInOffice* (from false to true) usually coincides with the transition period of *TempCup* (from hot to cold), the user is more likely to return to the office before the cup becomes cold. Furthermore, if the user does not return to his/her office within the transition period of *TempCup* (from hot to cold) a security-warning message should be raised.

With regard to the initial stage of interpretation based on the context history (i.e. insufficient context history to induce rules or probabilities for the interpretation) an approach based on predefined rules and multi learning algorithms could prove the most effective one. At this initial stage, the decision on the security level can be made by the predefined rules until enough situations are accumulated in the context history. However, in general, the rules themselves could be changed gradually. These changes can be come about without human intervention by adopting the decision tree method under the multi-layered learning design.

Through our examination of a learning system (based on Predefined Rules and a Single Learning Algorithm) we discovered a number of issues that will motivate our future work in this area.

- How can the uncertainty aforementioned be reduced? We need to investigate other machine learning methods or some supplementary process for adopting the probabilistic approach.
- How can the system explicitly understand and reflect the user's feedback on its future behaviour?
- How can the temporal elements be considered with context?
- In the long run, the life style of the user might be changed. How can the old context history be excluded from the process of interpretation in order to reflect only the recent patterns of behaviour?
- How can the final or the intermediate results be represented and updated in a user model for reuse?

5. Related Works

Several applications developed by Anhalt et el. [13] can minimise distractions in a pervasive computing environment, based on context-awareness. Firstly, the PHD (Portable Help Desk) application is designed to deliver information needed to the user in a proactive manner. For example, the PHD can suggest a nearby printer when the user begins a print job. Secondly, the Matchmaker application determines the most appropriate expert for the user's problems by considering experts' skills, availability, and distance from the user. Finally, context-aware agents deliver relevant information to the user according to the situation of the user and the priorities of each event in the user's calendar.

Two proactive model-based systems were suggested by Selker and Burleson [14]. Firstly, the COACH (COgnitive Adaptive Computer Help) system provides adaptive help to the user based on his/her level of skills, namely novice, intermediate, and professional. Secondly, the Music Ball system is designed to examine the potential of creating a relationship between the computer and the user. In more detail, the system continuously models the actions and preferences of a ball-user and provides sounds based on the user model.

Mtichell et al. [7] provided the CAP (Calendar APprentice) application that helps a user with scheduling calendars based on the user's scheduling preferences. In particular, through the design of the CAP, they explored the potential of machine learning methods for the implementation of personal software assistants. The CAP runs a decision tree learning algorithm on the past calendar information at each night, in order to refine the set of rules that will be used to provide advice on the following day.

The aforementioned applications can be considered as various cases that realise the concept of our PDS whose aim is to provide intimate and proactive personal assistants. Firstly, PHD and Matchmaker provide a kind of "rule-based proactive adaptation" based on current context. Secondly, COACH and Music Ball utilise user modelling techniques in order to provide adaptations based on, for example, the characteristics and preferences of the user. Finally, CAP employs a machine learning algorithm in order to provide proactive suggestions on the user's scheduling calendars. All these applications demonstrate part of our approach, namely context-awareness, user modelling, or machine learning. However, we argue that the synergistic combining of these distinct techniques is required in order to fully exploit the value of context history and support the development of intimate and proactive personal assistants.

6. Conclusion

This paper has explored the potential of utilising machine learning techniques in order to obtain higher levels of context interpretation based on context history. The security guard scenario was considered in order to examine two types of proactive behaviour, namely "proactive modelling-based adaptation" and "proactive rule-based adaptation". In order to examine how the two types of proactive behaviour can be implemented, three types of learning system were designed. The design for the use of predefined rules and a single learning algorithm was examined with the Naïve Bayesian Classifier. Under this design, we explored the exceptional situations in conjunction with predefined general rules. We also considered how an explanation for proactive behaviour could be provided to the user. However, in order to utilise the result from probabilistic learning methods, the consequence of the uncertainty must be further investigated.

To summarise, through the conceptual design of our PDS and the design for calculating the level of security risk, the following key issues arise:

i. Proactive behaviour can be broadly categorised into two types: "proactive modelling-based adaptation" and "rule-based proactive adaptation"
ii. Proactive modelling-based adaptation can provide dynamic adaptations in an environment where the predefined rules need to be adjusted according to changes of the user's life style or changes in the situation itself.
iii. Utilising context history together with user modelling and machine learning techniques is a novel approach for inferring patterns from the user's behaviours that is implicitly contained in context history. Such an approach has the potential to support dynamic adaptations.
iv. The provision of an explicit and understandable explanation for a proactive behaviour to the user is desirable in order to encourage a trust relationship between the user and the context-aware system. This raises an important requirement on the design of learning systems including the selection of appropriate learning methods.
v. We believe that a design based on the use of predefined rules and multi learning algorithms could be effective at both an initial stage of learning (i.e. when little context history exists) and also during later stages when it is necessary to support dynamic adaptations.

In the near future, we intend to experiment further with our design based on the use of predefined rules and multi learning algorithms. We also intend to look more closely into the support required for determining and exploiting patterns of behaviour based on temporal transitions relating to context history.

Reference

1. Dey A.K. and G.D. Abowd (2000) The Context Toolkit: Aiding the Development of Context-Enabled Applications. Workshop on Software Engineering for Wearable and Pervasive Computing, Limerick, Ireland.
2. Salber D. and G. D. Abowd (1998) The Design and Use of a Generic Context Server, Perceptual User Interfaces Workshop (PUI '98), San Francisco CA, pp. 63-66.
3. Weiser M. (1993) Some Computer Science Issues in Ubiquitous Computing, Communications of the ACM, Vol 36(7).
4. Kay A. (1991) Computers, Networks, and Education, Scientific American, September, pp 138 – 148.
5. Tesler L. (1991) Networked Computing in the 1990's, Scientific American, September, pp 86 – 93.
6. Weiser M. (1991) The Computer for the 21st Century, Scientific American, September, pp 94 – 104.
7. Mitchell T. M., R. Caruana, D. Freitag, J. McDermott and D. Zabowski (1994) Experience With a Learning Personal Assistant, Communications of the ACM, 37(7).
8. Mitchell T. M. (1997) Machine Learning, McGraw-Hill.
9. De Bra P. and L. Calvi (1998) AHA! An open Adaptive Hypermedia Architecture, The New Review of Hypermedia and Multimedia, Vol 4, pp 115-139.
10. Pohl W. (1996) Learning About the User – User Modeling and Machine Learning, Workshop in ICML'96 on Machine Learning meets Human-Computer Interaction.
11. Billsus D. and M. Pazzani (1997) Learning Probabilistic User Models
12. Ruvini J. D. and C. Fagot (1998) IBHYS: A New Approach to Learn Users Habits, Proceedings of ICTAI'98, IEEE Computer Society Press.
13. Anhalt J., A. Smailagic, D.P. Siewiorek, F. Gemperle, D. Salber, S. Weber, J. Beck and J. Jennings (2001) Toward Context-Aware Computing: Experiences and Lessons, IEEE Ingelligent Systems, Vol 16 (3), pp 38 – 46.
14. Selker T. and W. Burleson (2000) Context-aware design and interaction in computer systems, IBM System Journal, Vol 39 (3&4), pp 880 – 891.
15. Byun H.E. and K. Cheverst (2001) Exploiting User Models and Context-Awareness to Support Personal Daily Activities, Workshop in UM2001 on User Modelling for Context-Aware Applications, Sonthofen, Germany.
16. Lamming M. and M. Flynn (1994) "Forget-me-not" Intimate Computing in Support of Human Memory, Symposium on Next Generation Human Interface, Meguro Gajoen, Japan.
17. Dey A.K. and G.D. Abowd (2000) CyberMinder: A Context-Aware System for Supporting Reminders, Symposium on Handheld and Ubiquitous Computing, Bristol, UK.
18. Abowd G.D. and E. D. Mynatt (2000) Charting Past, Present and Future Research in Ubiquitous Computing. ACM Transactions on Computer-Human Interaction, Special issue on HCI in the new Millenium, Vol 7(1), pp 29-58.
19. Cheverst K., N. Davies, K. Mitchell and P. Smith (2000) Providing Tailored (Context-Aware) Information to City Visitors, Conference on Adaptive Hypermedia and Adaptive Web-based Systems, Trento.
20. Lesser V., M. Atighetchi, B. Benyo, B. Horling, A. Raja, R. Vincent, T. Wagner, P. Xuan and S. XQ. Zhang (1999) The Intelligent Home Testbed, Autonomy Control Software Workshop.
21. Gellersen H.W., M. Beigl and H. Krull (1999) The MediaCup: Awareness Technology embedded in an Everyday Object, Symposium on Handheld and Ubiquitous Computing, Karlsruhe, Germany.
22. Pascoe, J., "Adding Generic Contextual Capabilities to Wearable Computers", In the Proceedings of the 2nd IEEE International Symposium on Wearable Computers (ISWC'98), pp. 92-99, Pittsburgh, PA, IEEE. October 19-20, 1998.
23. Schmidt, A., Beigl, M. and Gellersen, H. W., "There is More to Context than Location", In Interactive Applications of Mobile Computing, Rostock, Germany, 24-25, November 1998.
24. Chen, D., Schmidt, A., Gellersen, H.W., "An Architecture for Multi-Sensor Fusion in Mobile Environments", In Proceedings International Conference on Information Fusion, Sunnyvale, CA, USA, July 1999.