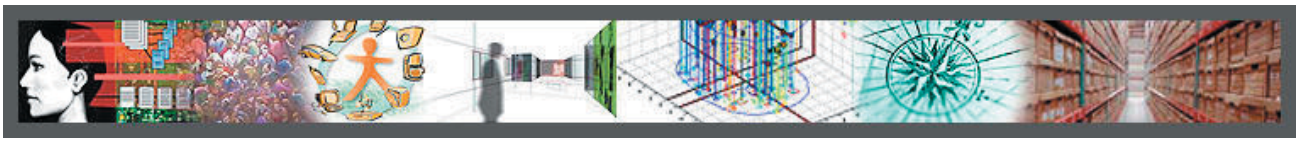
PROCEEDINGS

ECHISE2005

1st International Workshop on Exploiting Context Histories in Smart Environments

3rd International Conference on Pervasive Computing - PERVASIVE 2005 www.pervasive.ifi.lmu.de



WORKSHOP

May 11, 2005

at Munich, Germany

ORGANIZERS

Thorsten Prante
Fraunhofer IPSI, Darmstadt, Germany

Brian Meyers
Microsoft Research, Redmond, WA, USA

Geraldine Fitzpatrick
University of Sussex, Brighton, UK

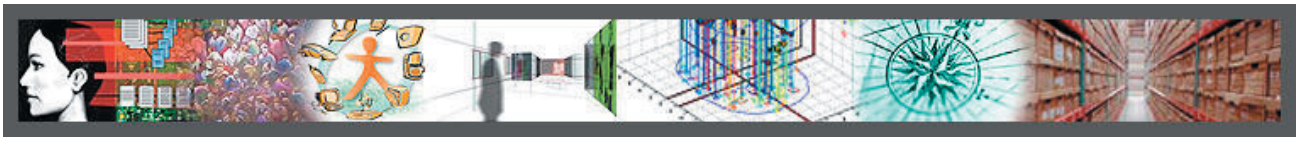
Lonnie D. Harvel
Georgia Institute of Technology, Atlanta, GA, USA

http://www.ipsi.fraunhofer.de/ambiente/echise2005

ECHISE2005

1st International Workshop on Exploiting Context Histories in Smart Environments

3rd International Conference on Pervasive Computing - PERVASIVE 2005 www.pervasive.ifi.lmu.de



CALL FOR PAPERS

http://www.ipsi.fraunhofer.de/ambiente/echise2005

Organizers

Thorsten Prante[1], Brian Meyers[2], Geraldine Fitzpatrick[3], Lonnie D. Harvel[4]

[1] Fraunhofer IPSI, Darmstadt, Germany
[2] Microsoft Research, Redmond, WA, USA
[3] University of Sussex, Brighton, UK
[4] Georgia Institute of Technology, Atlanta, GA, USA

Motivation

We are going to experience the diffusion of ubiquitous and pervasive computing technology into our everyday environments. The resulting interconnected collections of computational artifacts have to exhibit smart and coherent behavior in support of the users' activities and tasks across artifacts/devices, in order to be perceived as smart environments on the users' side.

As interactions tell stories, a promising approach to enhance the user experience in smart environments is to exploit recorded histories of the users' interactions in context (context histories for short). These histories can be used to support ubiquitous/pervasive computing applications with an enhanced understanding of the users' activities and interactions as they expand and develop over time. In addition, current contextual data is often imperfect and noisy and can be smoothed from the use of histories.

In more traditional computer use, interaction histories have been in use for quite some time. Interaction logs of evaluation tasks serve a detailed analysis. Histories of visited web sites are used to support navigation. Online shops offer recommender functionality based on interaction logs. Finally, interaction histories are sometimes used to infer user tasks. These and other applications can be enriched by integrating context sensing from even more sensors into the physical and digital world, e.g., also spanning across devices.

Exploiting context histories may, for example, result in less distractions, more adapted and coherent behavior of multi-computer setups, and better support in information management and search. The user could end up with a more supportive and comforting environment. At the same time, however, the collection, storage, management, and exploitation of context histories is a delicate issue, as privacy, informational self-determination, and data security are touched. In smart environments, there will be multiple users interacting with multiple and heterogeneous devices, some of them with little storage and computing power. How can we collect all the information needed? How does the data get synchronized between devices? And how does the user control and feel comfortable with that level of data being stored?In this workshop we want to bring together researchers to discuss the trade-offs of exploiting context histories for supporting user experiences with applications in smart environments. Several systems and approaches have already been proposed. We will set out to shape a common vision for this emerging field by identifying common requirements and issues.

Expected Audience

The workshop is directed towards researchers and practioners who are both interested in dealing with interaction histories and pervasive computing. 10 to 15 participants will be invited based on a position paper submitted prior to the workshop.

Four to six graduate students will also be invited to participate in the workshop. Students are not required (but allowed) to submit a position paper. Students should, however, submit a two-page paper outlining their research interests and their motivation to participate in the workshop.

Submissions

The position papers are expected to contain a clear statement about added-values vs. risks of exploiting context histories to enhance user experience in ubiquitous/pervasive computing environments. They should clearly state their contribution and identified open challenges.

Each position paper should be around three to four pages according to the ACM SIGCHI formatting (http://www.sigchi.org/chipubform). Longer submissions will be considered as well. A submission should be complemented by a short bio of the author/s. Please email your submission to Thorsten Prante.

Topics

Possible topics for submissions must relate to context histories and include (but are not limited to):

- User activities that benefit from context histories (personal and group support)
- Detection and exploitation of relationships among information and context components
- Issues of inferring activities from context histories
- Issues of determining appropriate levels of granularity for working with context histories
- User experience of temporal ambient displays and other visualizations
- Issues of context recording and playback
- Enhanced situated support for user activities across physical and digital worlds
- User ability to author an experience
- New forms of support for search, navigation, and orientation
- Interactional information and knowledge management in context
- Aspects of sharing information from or based on context histories
- Support for reflective activities
- Summarization of ever increasing datasets
- How to evaluate such systems from a user perspective
- Combining multiple interaction histories
- Appropriate context modelling
- Storage, management, and dissemination of information from context histories
- Appropriate representation of information from context histories
- Smoothing of context data
- Social implications and social protocols
- Privacy, informational self-determination, and data security issues

Important Dates

February 23, 2005	Submission deadline
March 8, 2005	Notification of acceptance
April 15, 2005	Camera-ready version due
May 11, 2005	Workshop takes place in Munich, Germany

Contact

Thorsten Prante
thorsten.prante@ipsi.fraunhofer.de

Contents

Talks

Page

Supporting Work Activities

Context Histories, Activites and Abstractions: Ubiquitous Computing Support for Individual and Collaborative Work 1
Stephen Voida, Elizabeth D. Mynatt (Georgia Institute of Technology, USA)

Managing Project Contexts: Interaction History as a Resource 5
Victor Kaptelinin (Umeå University, Sweden)

Transparent Interaction, Dynamic Generation: Context Histories for Shared Science 11
mc schraefel, Sacha Brostoff, Ray Cooke (University of Southampton, UK)
Robert Stevens, Andrew Gibson (University of Manchester, UK)

Determining Activity, Situation and Situation-Relevance

Using Context History for Data Collection in the Home 17
Daniel H.Wilson (Carnegie Mellon University, USA), Danny Wyatt (University of Washington, USA), Matthai Philipose (Intel Research Seattle, USA)

Situation Determination with Distributed Context Histories 21
Graham Thomson, Paddy Nixon, and Sotirios Terzis (University of Strathclyde, UK)

Immune Inspired Context Memory 26
Philipp H. Mohr, Jon Timmis, Nick Ryan (University of Kent, UK)

Delivering Services

Context Prediction based on Context Histories: Expected Benefits, Issues and Current State-of-the-Art 31
Rene Mayrhofer (Johannes Kepler University Linz, Austria)

A Stochastic Approach for Creating Context-Aware Services based on Context Histories in Smart Home 37
Hua Si, Yoshihiro Kawahara, Hiroyuki Morikawa, Tomonori Aoyama
(The University of Tokyo, Japan)

Building a Personal Memory for Situated User Support 43
Michael Schneider, Mathias Bauer, Alexander Kröner
(German Research Center for Artificial Intelligence DFKI, Germany)

MoBe: Context-Aware Mobile Applications on Mobile Devices for Mobile Users 49
Ivan Scagnetto, Andrea Selva, Luca Vassena, Paolo Zandegiacomo Riziò
(University of Udine, Italy)

Exploring Trails and Places, Alone and Together

The Geographic Context Browser 55
John Krumm (Microsoft Research, USA)

Sharing Photos and Recommendations in the City Streets 59
Marek Bell, Matthew Chalmers, Barry Brown, Ian MacColl, Malcolm Hall, Paul Rudman
(University of Glasgow, UK)

Sharing Context History in Mobile, Context-Aware Trails-Based Applications 64
Mike Spence, Cormac Driver, Siobhán Clarke (Trinity College Dublin, Ireland)

Posters

Page

Location Prediction, Knowledge Rediscorvery, and Self-Surveillance

Next Location Prediction Within a Smart Office Building 69
Jan Petzold, Faruk Bagci,Wolfgang Trumler, Theo Ungerer
(University of Augsburg, Germany)

Tracking Personal Histories for Knowledge Discovery 73
Dennis P. Groth (Indiana University, USA)

Phsycial WorkPace: A Case Study on Exploiting Context Histories
in Personal Healthcare Domain 77
Yuechen Qian (Eindhoven University of Technology, The Netherlands)

ShareAware: An Interactive Pervasive System to Promote Awareness
of Workstation Ergonomics 82
Anurag Sehgal, Patray Wing Lam Lui (Interaction Design Institute Ivrea, Italy)

Theory and Wonder

Integrating History and Activity Theory in Context Aware System Design 87
Manasawee Kaenampornpan, Eamonn O'Neill (University of Bath, UK)

Wonder Objects – Magic and Interactive Storytelling, 91
Tarun Jung Rawat (Interaction Design Institute Ivrea, Italy)

Configuration, Privacy, and Conflict Resolution

Exploiting Context Histories in Setting up an E-Home 97
Johannes Helander (Microsoft Research, USA)

Privacy Enhanced Active RFID Tag 100
Shingo Kinoshita, Miyako Ohkubo, Fumitaka Hoshino, Gembu Morohashi,
Osamu Shionoiri, Atsushi Kanai (NTT, Japan)

Conflict Resolution Method Utilizing Context History for Context-Aware Applications 105
Choonsung Shin, Woontack Woo (GIST U-VR Lab., South Korea)

Proceedings of ECHISE'05, held in Conjunction with PERVASIVE'05, May 11, 2005, Munich, Germany.

Context Histories, Activities, and Abstractions: Ubiquitous Computing Support for Individual and Collaborative Work

Stephen Voida & Elizabeth D. Mynatt
GVU Center, College of Computing
Georgia Institute of Technology
85 5th Street NW, Atlanta, Georgia 30332-0760 USA
{svoida, mynatt}@cc.gatech.edu

INTRODUCTION

As ubiquitous computing technologies find their way into widespread use and become an "invisible" and pervasive part of users' everyday practices, the relationship that users have with these technologies will begin to change. While users may have been content to adapt their practices to match the information management strategies and "application-document" models imposed on them by computers (and their designers) in the past, they will be less willing to do so as computers find their way into more aspects of everyday life and mediate more of our human-human interactions. Gay and Hembrooke have noted a corresponding shift in the language used by HCI practitioners—where *user-centered design* used to be the touchstone of the field, the ideas of *activity-* and *context-centered design* are becoming increasingly prevalent [4].

The workplace is a particularly interesting setting for studying this transition. Although the desktop computer is a long-established fixture in the office, mobile phones and networked devices like the RIM Blackberry have, for many workers, become as common and just as indispensable. The proliferation of Web-based corporate applications, virtual private networks, and VOIP telephony has extended the boundaries of the traditional workplace so that work now occurs in many non-traditional locations—and "on the go." Furthermore, these new technologies, in many cases, have not *replaced* existing technologies so much as they have served to *augment* them; the role of each technology is constantly changing, but the overall complexity of the workplace is, in general, on the rise.

As the *amount and diversity of incoming information* confronting knowledge workers steadily increases, the *devices used to carry out work* multiply, and the *locations in which work takes place* become more varied, more traditional computer-based practices for organizing and managing work begin to break down. Email is the most common example of this trend—it is widely acknowledged that email has become an incredibly overloaded medium, serving not only as a means for communication, but for coordination, scheduling, task-awareness, organizational memory, document sharing and version control (to name just a few) [2].

An increase in the amount of contextual information collected in the workplace and available to knowledge workers can be (and in fact *is*) part of this problem: it is just that much more information to be managed. However, it can also be an asset for helping users to maintain an overall awareness of their work environment, their ongoing work tasks, and the state of their collaborations with others, as well as a memory aid in task resumption. Our research has focused on the iterative development of computing systems that support these goals, based on models of activity created by compiling many sources of virtual and physical context. Such systems provide a structured environment that serves to organize work artifacts and context in a manner more consistent with knowledge workers' actual work practices.

Our research program lies at the intersection of two major bodies of research: activity-based computing and context-aware computing. Several field studies on the role of tasks and activity in the workplace have recently been published (e.g., [1, 5]) and some initial activity-based computing prototypes have been developed (e.g., [8]). Other research has focused on how context can be utilized as a part of existing work practices, most commonly as a tool for awareness and interruption management (e.g., [3]). Our initial explorations have been focused on investigating the role of activity modeling, peripheral displays, and integrated context-aware frameworks in supporting individual work. We are interested in expanding the scope of our inquiry to explore how adding collaboration support changes the requirements for activity- and context-aware systems.

Activity and Context in the Kimura System

Our prototype system, Kimura, was developed to help us understand how activity models, peripheral displays, and context-awareness could be used to support task-awareness and multitasking in knowledge work [7]. The Kimura prototype combines a desktop computer running a custom virtual desktop manager with an electronic whiteboard and context-aware infrastructure. As in previous systems like Rooms [6], users create virtual desktops on the computer to separate and organize their various work activities. Kimura builds a model of activity based on the "virtual context" of users' interactions with the desktop computer and virtual window manager. It then integrates other virtual and physical context sensed by the context-aware infrastructure into the model. We call the resulting clusters of

Figure 1. The Kimura system, including a desktop component, two interactive peripheral displays with electronic whiteboard capabilities, and a third non-interactive peripheral display. The images projected on the electronic whiteboards are *montages*, representations of activity that integrate history and context information.

computational artifacts and contextual cues *working contexts*, and display a representation of each, called a *montage*, on the electronic whiteboard. Users can view the whiteboard as a passive peripheral display and monitor the state of all ongoing work activities. They can also interact with the whiteboard directly to annotate, organize, and switch among working contexts.

Kimura's integration of virtual and physical context is unique [9]. The system creates a high-level framework of working contexts based on the virtual context—the user's manipulation of the virtual desktops and other interactions with the desktop computer—within which other virtual- and physical-context information is classified and interpreted. The system's context interpreter constantly updates the framework and the montage visualizations based on the stream of virtual and physical context captured by the context acquisition components.

This combination of interpreted context information provides detailed representations of each of the user's activities and is used to generate the montage visualizations displayed on the electronic whiteboard. The montage designs take advantage of several visualization techniques to express the working contexts' semantics. To show a summary of a working-context at a glance, montages contain thumbnail images of the user's desktop computer applications as well as icons representing relevant external context for each activity. These representations are also adapted to reflect the history of each activity, including the *relevance* of individual aspects (for example, time spent interacting with a given artifact or the inferred importance of a contextual cue) as well as their relative *recency* (providing a sense of the temporal evolution of the activity).

For a typical knowledge worker, Kimura might monitor a number of concurrent work activities, displaying a montage for each on the electronic whiteboard. Currently, these montages convey to the user what applications and documents have been accessed over the course of each work activity, which documents have been most important, any annotations the user has provided, and other context information about each activity such as whether colleagues affiliated with an activity are available for face-to-face collaboration (if they have been sensed in an office common area) or whether a print job relating to an activity has been completed and is awaiting retrieval.

CHALLENGES OF MODELING ACTIVITY AND CONTEXT HISTORIES FOR INDIVIDUAL WORK

Our experiences with the Kimura system confirmed our intuitions (and others') that activity can be a potentially powerful organizing principle for dealing with the increasing complexity of knowledge work. We feel that there are strong benefits for providing these representations of activity and context to both desktop and ubiquitous computing applications so they might assist the user in switching among ongoing tasks, creating new ongoing tasks that resemble previous ones, and maintaining an awareness of the tasks in which they are currently engaged. However, our initial models for representing activities and their associated context have proven to be somewhat inadequate for authentically modeling real-world work practices.

Models of activity should enable the expression of different classes of activities such as routine tasks and recurring tasks and different types of activities such as information analysis tasks and content production tasks. They should also be able to encode a broad range of affiliated context such as the location in which an activity was accomplished, the time (or frequency) at which it occurred, the individuals with whom the activity was carried out and what specific contributions each made. Systems implementing these sophisticated models will further benefit from maintaining details of activity and context over time, so that trends can be monitored and patterns detected, leading to representations of emergent behavior and enabling systems to suggest procedures or artifacts that have been useful in similar situations.

We envision a system like Kimura that enables users to demarcate their work activities and to organize their computational artifacts, relevant communications, colleague contact information, and personal reminders as an implicit part of their existing work practices (or with as little additional overhead as possible). This system should also allow users to search for past material using rich contextual cues as indices into past activities or recommend relevant information based on contextually-similar situations to ones the system has seen before.

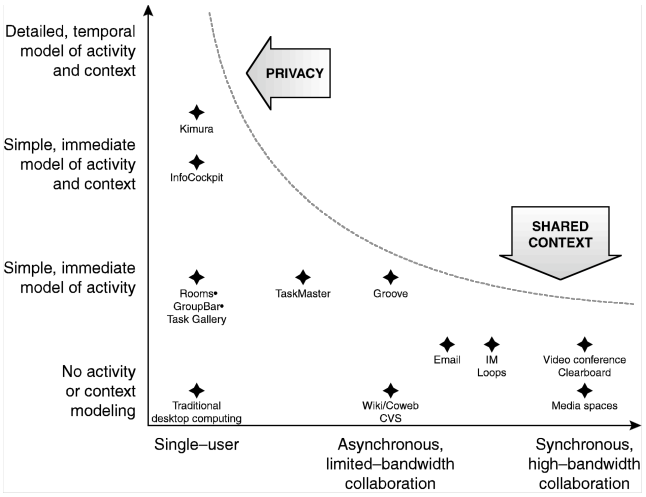


Figure 2. A design space for collaborative, activity- and context-aware applications. The large arrows indicate potential influences on the distribution of current applications within the design space.

***Challenge #1*: Integrating virtual and physical context to create a coherent model and history of user activity**

What are the critical characteristics of modeling activity and context over time? A successful model should reflect the findings of research on workplace activity and enable useful individual task management in ways not available with today's systems. However, maintaining a balance between flexibility and complexity will be important in order for applications to be able to utilize the modeled data and for users to be able to manage their representations.

CHALLENGES POSED BY COLLABORATION

However beneficial enhanced models of activity and context might be for supporting individual users, potential tensions exist *because* most activity-aware systems are targeted at individual use and many "real-world" knowledge work activities are inherently collaborative.

In order to understand how this tension has played out in existing systems, we constructed a design space illustrating the sophistication of activity-awareness and collaborative complexity of several commercial and research Ubicomp and workplace applications (Figure 2). Most of these systems cluster toward the individual-use, activity-aware portion of the diagram (the left-hand side) *or* toward the collaborative, non-/marginally-activity-aware portion of the diagram (along the bottom). We speculate that two forces may be acting on the position of systems in this design space: *privacy* and *inherently shared context*. The cluster of systems along the vertical axis may be constrained by concerns about privacy. These systems encode significant details about individual activity and context but are not equipped to represent these models appropriately for collaborative situations. In contrast, the cluster of collaborative applications along the horizontal axis may inherently convey some degree of shared context and activity-awareness as a by-product of the collaboration process. As a result, it may not be necessary for these applications to explicitly encode models of activity or context in order for the interaction to be successful in the context of working in a group.

***Challenge #2*: Addressing privacy concerns when collaborating with sophisticated models of user activity and context history**

As more and more detail about a user's actions and the context in which he or she carried out their work are captured and stored, the risk of having this potentially personal information inadvertently shared with others over the course of collaboration grows. Finding a balance between activity- and context-awareness and collaboration support requires difficult design trade-offs.

***Challenge #3*: Accommodating differences in granularity of activity specifications**

There will almost certainly be cases in which two users need to coordinate activities and context histories established independently. The way in which these models are specified will determine the complexity of "merging" the two models, particularly for cases in which the users conceive of and manage their activities at different levels of granularity. Resolving these differences elegantly is critical to these systems' success. The development of user interfaces and visual representations to ease merging models will likely be a critical area for research.

The Role of Abstractions

We are interested in developing tools that support all aspects of knowledge work, including individual work and collaboration. However, in order to do so, we need to find ways to overcome the potential privacy issues involved in sharing personal activity and context information, and, if possible, integrate the representations of shared context in the collaboration process itself, as do many existing tools.

We hypothesize that providing varying levels of abstraction in our activity and context histories can allow users to

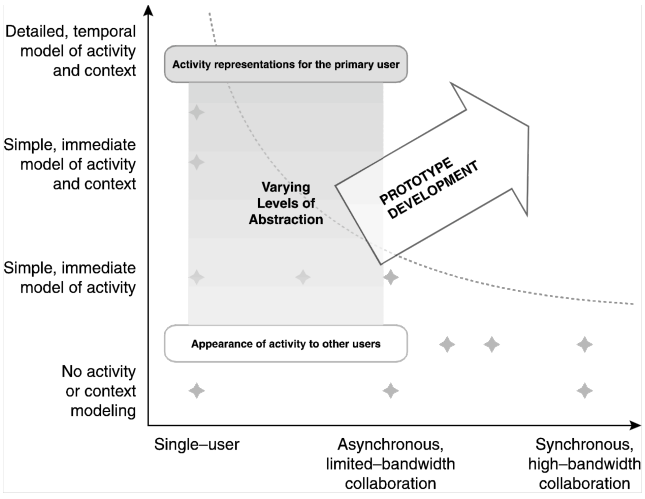


Figure 3. Our proposed work focuses on the development of systems that encode rich activity and context histories, but also provide user interfaces and representations for controlling abstractions, so that the systems can be appropriated for individual use *and* collaboration.

specify the level of detail most appropriate for a given situation: while collaborating with a particular group of colleagues, working in a specific location, or working on a particular device (Figure 3). We believe that this approach gives users the most flexibility, allowing them to take full advantage of activity- and context-awareness when working individually and providing them access to activity and context information when needed during collaboration.

Challenge #4: **Identifying critical characteristics of activity and context histories for which collaboration hinges on having the right abstraction(s)**

Abstractions will likely be more critical for some aspects of activity and context histories than others. Due to the potential complexity of these histories and the myriad ways abstraction could be used to limit the disclosure of personal information, identifying the information users are most interested in protecting—and to what degree that information needs to be aggregated, anonymized, or excluded from histories shared with others—will be critical in informing the design of appropriate abstractions.

Challenge #5**: Providing user interfaces to manage abstractions**

Users will likely need to provide some degree of direct control or fine-tuning over the abstractions used in a given situation. However, this requires imposing additional "meta-work" on top of the work practices users already have in place. What user interfaces are most appropriate for managing abstractions of activity and context histories? Are there instances in which implicit observation of existing work practices can be used to determine the appropriate abstraction to apply?

Challenge #6**: Examining the role of the user's location and the devices they use in selecting an appropriate level of abstraction for a given context**

Our initial explorations have taken advantage of a subset of Ubicomp technologies we felt most appropriate for integration into an individual's existing workspace. Can the virtual or physical context sensed using a broader range of devices (including those specifically designed to support collaboration) be used to reliably infer the level of abstraction most appropriate for a particular situation?

OBJECTIVES FOR THE WORKSHOP

We are looking forward to participating in the ECHISE2005 workshop since it appears be an ideal venue for us to refine and inform our intuitions about the challenges in designing these types of systems based on the research being carried out by others in the field.

We are particularly interested in discussing the methods that are being used to model complex activity and context histories in other systems, the means for abstraction that these other approaches employ, and the user interface conventions others have found successful for representing and providing user control over context histories.

AUTHOR BIOGRAPHIES

Stephen Voida is a Ph.D. student in the Georgia Institute of Technology's College of Computing and a member of the GVU Center and the Everyday Computing lab. His research interests include ubiquitous computing, technology in the workplace, and augmented environments. He received his M.S. in human-computer interaction from the Georgia Institute of Technology in 2001.

Elizabeth D. Mynatt is an associate professor in the Georgia Institute of Technology's College of Computing and the GVU Center. She directs the Everyday Computing research program within the Future Computing Environments group, examining the implications of having computation continuously present in many aspects of everyday life. Her research interests include exploring how to augment everyday places and objects with computational capabilities. She received her Ph.D. in computer science from the Georgia Institute of Technology in 1995.

REFERENCES

1. Czerwinski, M., Horvitz, E., and Wilhite, S. A diary study of task switching and interruptions, in *Proceedings of CHI 2004* (Vienna, Austria, April 2004), ACM Press, 175–182.
2. Ducheneaut, N. and Bellotti, V. Email as habitat: An exploration of embedded personal information management. *ACM Interactions, 8*, 5 (September–October 2001), 30–38.
3. Fogarty, J., Hudson, S.E. and Lai, J. Examining the robustness of sensor–based statistical models of human interruptability, in *Proceedings of CHI 2004* (Vienna, Austria, April 2004), ACM Press, 207–214.
4. Gay, G. and Hembrooke, H. *Activity-centered design: An ecological approach to designing smart tools and usable systems*. MIT Press, Cambridge, Massachusetts, 2003.
5. Gonzàlez, V.M. and Mark, G. "Constant, constant multi-tasking craziness": Managing multiple working spheres, in *Proceedings of CHI 2004* (Vienna, Austria, April 2004), ACM Press, 113–120.
6. Henderson, J.D.A. and Card, S.K. Rooms: The use of multiple virtual workspaces to reduce space contention in window-based graphical user interfaces. *ACM Transactions on Graphics, 5*, 3 (July 1986), 211–241.
7. MacIntyre, B., Mynatt, E.D., Voida, S., Hansen, K.M., Tullio, J., & Corso, G.M. Support for multitasking and background awareness using interactive peripheral displays, in *Proceedings of UIST '01* (Orlando, FL, November 2001), ACM Press, 41–50.
8. Smith, G., Baudisch, P., Robertson, G., Czerwinski, M., Meyers, B., Robbins, D. and Andrews, D. GroupBar: The TaskBar evolved, in *Proceedings of OZCHI '03* (Brisbane, Australia, November 2003), University of Queensland, 34–43.
9. Voida, S., Mynatt, E.D., MacIntyre, B. and Corso, G.M. Integrating virtual and physical context to support knowledge workers. *IEEE Pervasive Computing*, *1*, 3 (July–September 2002), 73–79.

Managing project contexts: Interaction history as a resource

Victor Kaptelinin
Department of Informatics
Umeå University
901 87 Umeå, Sweden
+46 90 786 5927
vklinin@informatik.umu.se

ABSTRACT
The paper deals with exploiting the potential of interaction histories for managing multiple project contexts in both traditional and smart environments. Mapping interaction histories to specific projects is proposed as a way to make interaction histories a useful resource for supporting continuous, coordinated work on a set of projects over time and distributing resources across contexts and devices. The proposed approach is illustrated with a simple example of using project-specific interaction histories for synchronizing work between a personal computer and a mobile device. Implications of the proposed approach to design of smart environments are discussed.

Keywords
Interaction history, project context, distributed work

INTRODUCTION
Even the modestly smart environments of today, featuring, for instance, automatic doors or sensor-based lighting, may cause problems for people in the environments by imposing excessive constraints, creating uncertainty, and misinterpreting user intentions. If car doors unlock automatically when the owner is approaching, how can one check if the locks work properly?

When environments become more "intelligent," the risks of causing mismatches between user's and system's models of interaction are likely to increase. Development of new interaction techniques capable of minimizing such risks is considered a key issue in design of smart environments [1].

This paper argues that exploiting interaction histories in smart environments can be facilitated by allowing the users themselves indicate (implicitly or explicitly) what their goals are. More specifically, it is suggested that providing support for selecting the currently active project – a relatively long-term sequence of tasks, subordinated to a higher-level goal, distributed in time and place, and often interrupted – can help utilize information contained in interaction histories and provide support to people acting in smart environments. The analysis in the paper is based on experience of employing interaction histories in a traditional desktop environment. Capitalizing on this experience, the paper makes an attempt to address issues related to smart environments.

The rest of the paper is organized as follows. The next section identifies the need to cope with the enormous volume of data that can potentially be included in interaction histories, in both traditional and smart environments. After that a number of possible ways to make interaction history data more manageable are discussed, including mapping events in interaction history to user's projects. Then a simple example of utilizing project-related interaction histories to support work distributed between several computing devices is presented. The paper concludes with a reflection on the implications of the proposed approach for creating smart environments.

INTERACTION HISTORIES: LIMITED YET ABUNDANT
Preserving and examining the traces human activities leave in the physical world may require a considerable effort. By contrast, traces left in virtual environments allow for relatively effortless storage and analysis. Given enough memory space and processing power, information technologies can record user inputs (or other external inputs), system events, and store them in the form of automatically created interaction histories. The potential of interaction histories for supporting the user was recognized by researchers and practitioners quite early. In the field of Human-Computer Interaction (HCI) interaction histories have been an important research issue for over a decade. For instance, a panel organized at the CHI'94 Conference [10] identified main functions of interaction histories in interactive systems and formulated an agenda for future studies in that area. In software development interaction histories have been practically employed, in one way or another, in a wide range of computer applications and systems [14].

Arguably, however, both research and practical applications of interaction histories are still in their infancy. For the most part, researchers and practitioners focussed so far on relatively simple and obvious uses of interaction histories. The list of issues waiting to be properly addressed, indicated in this workshop's call for papers [13], testifies that interaction histories remain a largely untapped resource in HCI.

There are at least two reasons why interaction histories have been difficult to study and use in traditional HCI. First, the possibilities for collecting informative interacting histories are rather limited. Recording low-level events, such as keystrokes or mouse clicks, is a relatively simple task but inferring user actions, -- and objects employed in the actions, -- from the low-level evens is often problematic. Some programs, such as Microsoft Office ® applications, generate higher-level events and thus support collection of informative interaction histories. However, many programs do not provide such support. In addition, very few systems automatically capture user actions in the physical world, such as, talking to a colleague during lunch or placing a carbon copy of a document in a physical folder. Therefore, an important part of users' everyday activities is not represented in interaction histories.

Second, even though interaction histories are limited, they can be excessively large. According to our experience [8, 9], recording interaction histories generates volumes of data, which makes it impossible for users to keep track of unprocessed histories. To make use of interaction histories users have to rely on representations produced by the system. Currently, little is known about how to present interaction histories to the user so that they are helpful rather than confusing.

Therefore, interaction histories are at the same time limited and abundant. Moving from traditional computer use to smart environments alleviates the first of these problems. Sensor technologies open up radically new possibilities for capturing human interaction with the world.

However, the second problem – abundance – is likely to get worse. The sheer amount of data generated by smart environments can be overwhelming. Even the most advanced storage devices can be insufficient for storing all that data. Therefore, the question of how much the system should remember remains open [15].

The volume and diversity of data in smart environments also present a problem for analysis of the data. The fact that the data is analyzed automatically does not by itself eliminate the problem. If people who create or otherwise control technology have a vague or unrealistic idea of how interaction history data can support interaction in principle, no processing power can rectify that.

BREAKING DOWN THE FLOW

There are two main ways to reduce the complexity of a recorded interaction history and make it more manageable. The first way is to summarize the information contained in the history, for instance, with tables, charts, or timelines, displaying the frequency, aggregate time spent, or distribution of certain types of actions or certain objects. Such representations of an interaction history in general could be useful, for instance, for reflection or accounting. Summarized representations can also be used automatically. For instance, if it is established that at a particular time people form lines at a certain ATM, at that time a sign could display information pointing out to other available ATMs in the area, while at other times the same sign can display different information.

The second way to make information contained in an interaction history more practically useful is to process the information and transform it into a form relevant to the task at hand (cf. [8]. Analysis of literature reveals several strategies employed to relate interaction histories to user tasks: (a) identifying patterns of co-occurring objects, (b) selecting a sub-set of history on the basis of formal criteria, (b) mapping to objects, and (d) mapping to projects.

Identifying patterns of co-occurring objects includes selecting an object, such as an email address [5.7] and detecting other objects that appeared in an interaction history concurrently with the selected one. The structure of the associations created, for instance, by applying cluster analysis techniques, can be visualised as a configuration of nodes linked to the selected object and to each other. This type of analysis opens up a possibility for a user to find objects relevant to the task at hand by following their links to other objects. The user can start with an available object, browse through its links (if necessary, selecting an associated object and exploring, in turn, its links, etc.) and eventually find relevant resources.

Selecting a part of interaction history on the basis of formal criteria is similar to using the "Find" function: the user can select a time period, type and name of objects, and so forth, to create a smaller-scale, more manageable subset of an interaction history. For instance, the user can single out events that took place last week, which involved using documents with "ECHISE" in their names. An example of selecting a subset of interaction history is creating a sub-stream in the Lifestreams system [6].

Mapping to objects is linking events in interaction histories to specific objects. It allows the user to see the history of actions with an object by simply selecting the object. This approach was employed, for instance, in design of educational technologies [14]. The history of actions carried out by a student with an object in a simulation environment can be viewed by other students, and thus support communication, reflection, and mutual learning,

Mapping to projects is linking events in interaction history to user's projects. We define projects as *higher-level, longer-term tasks*. Mapping events to projects allows the user to filter out irrelevant parts of interaction history and focuses only on relevant events when working on a project.

The rationale behind this approach is supporting users in managing projects. Since projects are carried out to attain higher-level goals, they are relatively independent from concrete information technologies. For instance, one can invite guests to a party via email, IM, SMS, phone, postcards, or face-to-face communication.

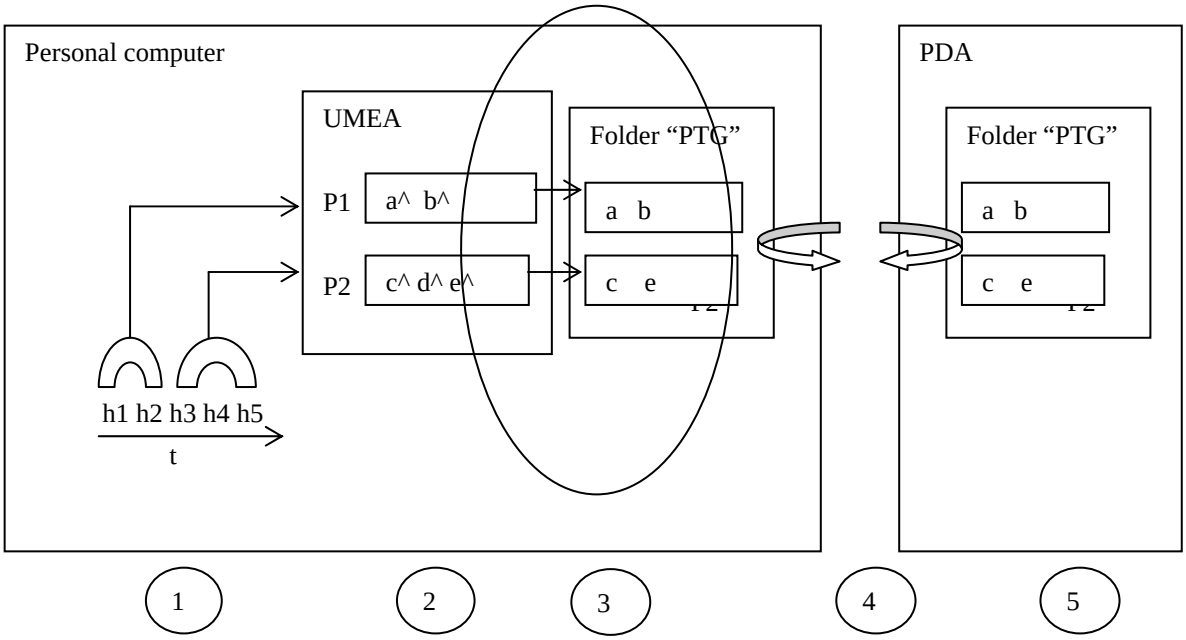


Fig. 1. "Packing for a trip": Copying files to PDA

Since projects are longer-term tasks, they are typically carried out in several sessions, distributed over time and intertwined with periods of work on other tasks. Therefore, working on a project requires: (a) ongoing coordination, making decisions about when to work on what project, (b) maintaining the continuity of working on a project despite pauses and breaks, and (c) integrating activities performed with various tools within one project. Empirical studies of computer users indicate that these problems are real.

Detailed, micro-level studies of the everyday use of information technology [2, 4, 9] revealed that people are constantly switching between different tasks. According to Czerwinski et al [4] "returned-to tasks," that is, tasks that tend to be resumed after an interruption, have a special status in the structure of user work. The study "… demonstrated that returned-to projects were more complex, on average, than short-term activities. These key projects were significantly lengthier in duration, required significantly more documents, were interrupted more, and experienced more revisits by the user after interludes." [4, p. 179]. At the same time, it was found that "… the reinstatement of complex, long-term projects was poorly supported by current software systems." [4, p. 175].

Mapping interaction histories to projects opens up possibilities to "stitch" separate sessions of working on a project into a coherent sequence of actions leading to the overarching aim of the project.

EXAMPLE: MANAGING TECHNOLOGICAL DISTRIBUTION OF WORK

Mapping interaction histories to projects can help maintain project coherence not only over time but also across various computing devices used within a project. Let us consider a simple example illustrating this claim.

In previous papers we presented a system named UMEA (User-Monitoring Environment for Activities) [11]. The system allows the user to define a set of projects and select one of the projects as active. The system monitors user actions and resources used within the active project, and automatically compiles project-related lists of resources. Entries in the calendar, notes, and "to do" lists are automatically linked to the active project, too. Therefore, when the user returns to a project by selecting it as active, the user gets convenient access to resources necessary for working on a project. At the same time, the user makes it possible for the system to update project workspace. An empirical evaluation of the UMEA system demonstrated that it helped users in managing their projects. At the same time, the evaluation identified possibilities for further improvement.

As a result of the empirical evaluation, the UMEA system was re-designed. One of the features added to the new version of the system (not yet reported) was intended to support distribution of work between several devices, for instance, between a desktop computer and a mobile device, such as a laptop, PDA, or smartphone. Mobile devices allow working on some tasks, such as reading and editing documents, when a regular personal computer is not available. However, limited memory space available on PDAs may make it impossible for the user to store all resources the he or she might possibly need. To support users in dealing with these problems the following feature was added.

Files related to different projects are automatically placed in different sub-folders.

The "PTG" folder serves for synchronization between the personal computer and the PDA (4). When the personal computer is synchronized with the PDA, using a standard synchronization feature of existing PDAs, resources selected by the user and stored in the "PTG" folder are copied to PDA's memory (5). If the user continues working on a project and creates new files or new versions of old files, these resources will be copied to the personal computer during the next synchronization session, again, using the standard functionality of existing handheld

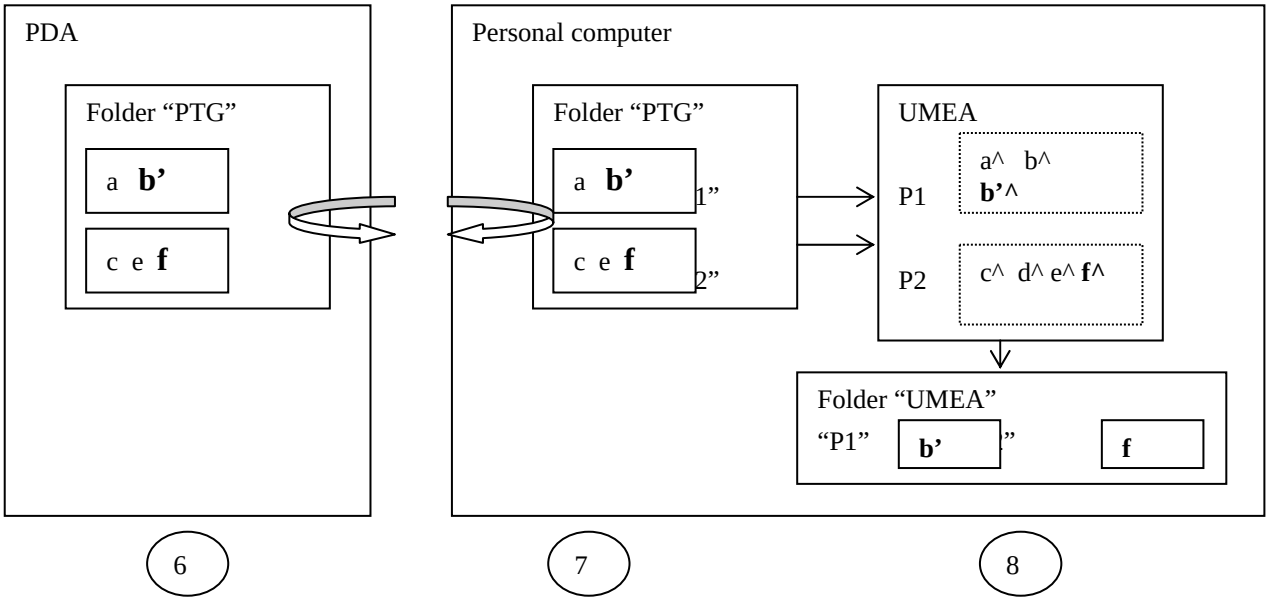


Fig. 2. "Unpacking": Adding new files from PDA to personal computer

The feature is schematically illustrated with Fig. 1. Via monitoring user actions (1) the system creates an interaction history, recorded as a sequence of events (h1-h5), where each event is an action carried out with a file (such as opening or saving). Events h1 and h2 are linked to project P1, while events h3-h5 are linked to project P2. By identifying files indicated in event descriptions the system links files **a** and **b** to project P1 and files **c, d**, and **e** to project P2 (2). The user can open the files from within the UMEA system by selecting a link to a file. The user does not need to know where a file is located. If the user wants to copy necessary files to a mobile device, he or she issues the "Project to go" command (3). The system displays a dialog window. The user browses through the files and indicates, which of them should be copied to the "PTG" folder (or any other folder selected by the user). Therefore, even though project-related files can be distributed all over the file system, the user can easily copy them to one folder.

devices.

In the next version of the UMEA system the "Project to go" feature is expected to be further advanced. Functioning of the prospective feature is shown in Fig. 2. New project-related files or new versions of existing files, created or copied by the user when working on the PDA (6) are added to a personal computer during next synchronization (7). These files are detected by the UMEA system and added to lists of files of their respective projects (8). In addition, the files are copied to appropriate project folders to make sure that they are not lost in the future.

The "Project to go" feature illustrates how the UMEA system uses interaction histories to distribute resources between a personal computer and a PDA.

To illustrate how a similar approach can be employed in a smart environment, let us consider the following imaginary scenario. When the user works on a project, the environment keeps track of using both virtual and physical

resources. The system can display, for instance, the list of books and papers related to the project, their locations, when they were used last time, and so forth. When the user prepares for a meeting, lists of project-related resources can help decide which papers should be taken to the meeting and where to find them. When several projects are discussed during the meeting and new documents are distributed to the participants, the smart conference room keeps track of which documents are used within which project. This information is transferred to the user's personal work environment (for instance, the user can download it to his or her PDA) and when the user comes back from the meeting with a bunch of papers, these papers are automatically added as new resources to their respective projects.

CONCLUSIONS

Interaction histories remain to be a largely unexplored resource in human-computer interaction. This paper discusses one particular approach to using interaction histories, that is, mapping interaction histories to projects to support managing multiple project contexts.

More specifically, the paper draws on the experience of employing interaction histories in traditional desktop environments and argues that a promising way to design smart environments is:

(a) letting people choose what they want to do rather than inferring user intentions from available data, and

(b) making sure, in a non-obtrusive way, that relevant resources are "ready to hand" when the user needs them.

According to Streitz and Nixon, a key issue in designing smart environments is "When does the system (or the infrastructure) try to predict the user's intentions and when are the users presented with choices?" [15]. The analysis in this paper allows to formulate two tentative guidelines addressing this issue. First, an articulation of user's intentions should preferably be a "by product" of attaining a meaningful goal. For instance, a user of the UMEA system may make a project active just to get an access to project resources. A by-product of that is making it possible for the system to map user actions to the project.

Second, even in cases when users intentions are inferred, the user should be able to control the system. In the second version of the UMEA system users can *link* resources to projects. Selecting a linked resource automatically makes the corresponding project active. In this case user's intention to switch to another project is inferred by the system. But it is the user, who determines how the system works. For instance, the user can unlink the resource. An elegant combination of system inference and user control is described by Cypher [3]. His Eager system suggests the next action when it recognizes a repetitive activity. However, it does not constrain the user. The user can continue working as usual and when he or she feels confident that the intention is recognized correctly, the user can let the system finish the task.

Of course, further work is needed to establish how/if the approach and guidelines presented in this paper can be applied in design of smart environments.

REFERENCES

1. Coutaz, J., Crowly, J. L., Dobson, S., Garlan, D. Context is key. *Communications ACM*, 48 (3), 2005.
2. Cypher, A. The structure of users' activities. In: D. Norman, S. Draper. (eds.) *User Centered System Design, New Perspectives on Human-Computer Interaction*. Lawrence Erlbaum Associates, 1986.
3. Cypher, Allen. "Eager: Programming Repetitive Tasks by Example". In Proceedings of CHI, 1991 (New Orleans, April 28 - May 2). ACM, New York, 1991.
4. Czerwinski, M., Horvitz, E., Wilhite, S. A diary study of task switching and interruptions. *Proceedings of the CHI 2004 Conference on Human Factors in Computing Systems* (Vienna, Austria, April 2004).
5. Farnham, S. Personal Map: Automatically modelling the user's online social network. *Position paper at the CSCW 2002 workshop "Redesigning email for the 21st Century"* (New Orleans, Louisiana, November 2002), 2002.
6. Fertig, S., Freeman, E., Gelernter, D. "Finding and Reminding" Reconsidered. *SIGCHI Bulletin*, v. 28, January 1996.
7. Fisher, D., Dourish, P. Social and temporal structures in everyday collaboration. *Proceedings of the CHI 2004 2004 Conference on Human Factors in Computing Systems* (Vienna, Austria, April 2004). ACM Press, 2004.
8. Fischer, G. Articulating the task at hand and making information relevant to it. Human-Computer Interaction, 16(2-4), 2001.
9. Gonzales, V., Mark, G. "Constant, constant, multi-tasking craziness": Managing multiple working spheres. *Proceedings of the CHI 2004 Conference on Human Factors in Computing Systems* (Vienna, Austria, April 2004). ACM Press, 2004.
10. Hill, W., Terveen, L. New uses and abuses of interaction history: Help form the research agenda. *Proceedings of the CHI'94 Conference on Human Factors in Computing Systems*, 1994.
11. Kaptelinin, V. UMEA: Translating interaction histories into project contexts. *Proceedings of the CHI 2003 Conference on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, April 2003). ACM Press, 2003.

12. Nardi, B. (ed.) *Context and consciousness: Activity theory and human-computer interaction.* Cambridge, Mass.: The MIT Press, 1996.
13. Prante, T., Meyers, B., Fitzpatrick, G. Harvel, L. D. *1st International Workshop on Exploiting Context Histories in Smart Environments. Call for papers.* 2005.
14. Shneiderman, B. (1999) *The Future of History*. Available at: http://www.cs.umd.edu/hcil/about/events/history-workshop/slides/Shneiderman/ index.htm
15. Streitz, N., Nixon, P. The disappearing computer: Introduction. *Communications ACM*, 48(3), 2005.

Author biography

Victor Kaptelinin is a Professor in the Department of Informatics, Umeå University. His current work deals with activity theory as a theoretical framework in Human-Computer Interaction, mobile technologies in education, and design of integrated digital work environments.

Transparent interaction; dynamic generation: context histories for shared science

mc schraefel, Sacha Brostoff, Ray Cooke
IAM Group
University of Southampton
Southampton, Hants, UK
http://www.ecs.soton.ac.uk
[mc, sb4, rc2] @soton.ac.uk

Robert Stevens, Andrew Gibson
BioHealth Informatics Group
School of Computer Science
University of Manchester
Manchester, UK
[robert.stevens,
Andrew.gibson]@manchester.ac.uk

ABSTRACT
Scientists who do *in silico* or computer-based experiments use general purpose computer tools, like Web browsers and word processors to carry out their tasks. As such, they have no formal file management support for collecting, coordinating, annotating and reflecting on their digital experimental traces. In this presentation we look at how we are exploring the use of implicit context histories to support scientists with both formal and everyday collaborations. We describe our goal to utilize the non-intrusive discovery and use of implicit contexts generated by task-based interactions in order to represent back, on demand, how one file or collection may be related to another. Such annotatable reports can then either be shared or used as inputs for further service requests for selected data.

Keywords
Transparent interaction, file management, metadata, semantic web.

INTRODUCTION
EScience is a new domain for HCI research. EScience seeks to use new networked computing opportunities such as the Grid to enable new science. [3] Part of these emerging requirements in this new field is to investigate ways to support a range of activities from the particular needs of scientific collaboration, to the requirements for demonstratable trustfulness of a system. One of the challenges in this space is to look at ways to support and enhance existing practice, as per Ubicomp's goals of transparent interaction [1, 2] in these rich lab-orineted environments, since much current practice is carried out with tools (from paper to mechanical devices to computer) which were not designed for data interchange or collaborative reflection. As a case in point, we have been looking at the practices of bioinformaticians, scientists who carry out their work almost exclusively *in silico* or on the computer, rather than *in vitro*, in the traditional wet lab. It would seem that in such an environment where work is already digital, integration and sharing of data would be less of a challenge than with their paper-bound colleagues. Alas, no. These disparate file traces have no medium through which they may be associated. Context histories, however, provide a possible vehicle for dynamic, sharable associations.

We have only recently completed our ethnographic studies and technologies review for the bioinformatics design space. In the following sections, therefore, we wish to report on an overview of these findings, the current design strategies based on them, their relation to context histories, and concerns surrounding the use/propagation of same.

Background: Experimental Recording in Bioinformatics
Bioinformaticians by way of background, are involved in molecular biological research. They run complex scientific experiments on myriads of biological data. Rather than running these experiments in the messiness of a traditional wet lab, their lab is generally a laptop computer connected to the Internet. This virtual lab is still, frequently, just as messy a space as their physical counterparts (see [14] for views of wet labs): digital files that are created in the heat of the experimental moment mayn't be saved with optimal names for later discovery. It is also up to the scientist to crack open a text editor in order to create annotations about a finding in progress. As has been shown elsewhere [15] copying data from the web into new files frequently leads to critical data, like descriptive names or originating URLs to be left off, making later recovery of information difficult to accomplish. Some bespoke services, such as myGrid, which run workflows of search patterns on gene databases have saved the scientist days of effort in having to run these web site crawls manually [16, 17] but the runs themselves still create legions of files associated with a given experiment which must be analyzed, assessed, and referenced as relevant or not.

As such, the recording of experiments is a largely ad hoc (or post hoc) and manual process, requiring the scientists to cadge together a variety of existing general applications (Web browsers, word processors, tools they may have written themselves for specific analytical tasks) to support their work. In other words, these new *in silico* based scientists do not have what their traditional wet lab colleagues routinely have to track the progress of an

experiment: they do not always have or use lab books. We have observed that many bioinformaticians do not use lab books; implicit notes are taken in the creation of a file store (folder names, dates, file names and readme files). In addition, if data are lost or uncertain, an experiment can be re-run simply in a manner not possible at the wet lab bench.

Lab books themselves, however, are not an optimal solution. Going back to paper in a digital field re-introduces the disadvantages of paper, the lack of sharability of results being key.

REQUIREMENTS

In observations of and meetings with bioinformaticians, it is clear that they would like a utility that would allow them to

- Generate dynamic reports referencing and linking to related files on a particular experiment, both the data and supporting material
- Allow multiple views on how one file relates to another
- Supports annotation of files by meaningful markers, both the for the biology, bioinformatics as well as the process of discovery and investigation itself
- Supports sharing a subset of these notes and files for collaboration, itself producing further annotations

The scientists have asked us for these types of controls not only in order to help them find previously potentially mislaid files (experiments can run for months or years), but also to help them share the state of their work, or subsets of it, readily with other collaborators.

They have also asked us to provide not only machine support for dynamic report generation, but human support, such as the ability to define a naming convention for a series of files and to have that convention (date, gene family for instance) applied automatically. Richer kinds of labels have also been requested, so that they can see at a glance what files are active which are potential candidates and which have been used and discarded.

Our frame of reference for these requirements has been to find a way to put some of the benefits of the lab book into the bioinformatitan's process. In particular, we wish to support the lab book's functionality to provide in one place a view on the processes and annotations on those processes associated with an experiment or collection of experiments, and the ability to browse through previous work. It is clear, however, that asking the scientist to carry out the file management tasks they would need to do to create these views manually is unacceptable. We also do not wish to ask them to change their favorite tools in order to use a "digital lab book" that would attempt to be part browser, part email client, part scrap book and part word processor. We would rather leverage the input/output created in using these tools and make such reports which reference this I/O available on demand.

(Implicit) Context Histories

The use of context histories is a means towards creating just such transparent, reusable tracking of associated information. In this case, we understand context histories to be the history of interactions traceable within the interactions with the computer which can be seen to be associated with a given experiment. We have been thinking of these as implicit or possibly latent context histories since they will be teased out from the history of input/output interactions logged in the system as files are created, manipulated and deleted. To support transparency, these histories will be made available on demand, linked to the appropriate files, and providing opportunities for annotation on the context as well as annotation of a specific artifact. These context histories can then be viewed from multiple perspectives, shared and altered by scientists with their communities to reflect on the progress of a study for feedback, or to share the evidence of a specific conclusion.

Our challenges are

- to derive the correct/required contexts from the available interaction history of a scientist's laptop,
- to provide appropriate forms of representation for viewing these histories along multiple perspectives.
- To annotate and/or tag the files in ways which support organization in these contexts
- To ensure that manual effort can remain at the level of a secondary task rather than be forced regularly into primary attention.

The last point in particular is inspired by concepts like marking menus [9] which support secondary interaction of tasks like copying or pasting by allowing a simple gesture to invoke the action anywhere on the screen rather than requiring a person to acquire a specific target, navigate the associated menu, and activate the command. We wish to support any required manual annotation of files in a similarly transparent, context-based approach. Our goal, however, is that we will be able to deduce sufficient value from a scientists' interactions that we will be able to build up a context history and use this for constructing appropriate associations. It will then be easier to subtract mis-additions or flag/annotate collected files than either to construct all contexts and additions manually.

RELATED WORK

Our approach is informed by three related efforts: innovative research in desktop replacement or desktop assistant models, virtual notebook applications and Semantic Web frameworks. We describe each in turn.

Desktop Replacements

The closest related work to the type of transparent interaction we are describing are desktop replacement systems which either replace or enchance the traditional desktop. Reikimoto's Timescape is perhaps one of the most

oft-sited examples of such a system. In Timescape, the paradigm of file-based hierarchies is changed to temporal views of spatially associated filed for exploring information contexts [12]. A person can therefore travel backwards or forwards in time to watch how an interaction with a file may have progressed.

Presto [5] is a java based networked desktop replacement, enabled by a sophisticated infrastructure to trap changes to documents/data, and which allows much greater flexibility in document organization than traditional hierarchical file systems. It interoperate with Solaris, Windows NT, and common applications like MS Word, and uses automated (through feature extraction) and manually generated attributes to group documents. It concentrates on dynamic reorganization of objects on the desktop, rather than generating a history that can be shared (although collections are shared). It has multiple inheritance - documents can appear in more than one category or collection. It has a centralized metadata store, that runs across the network extracting features from document contents and existing metadata (creation time, owner, filename, etc.) from where they are stored locally or on shared resources and visualizes them on each user's Presto desktop via an application called Vista. These documents can be launched and worked upon with the user's usual tools, but need to be manually associated with particular projects or categories.

Desktop Supplements for Context

As an alternate to desktop replacements, there are a set of applications which may be considered to be desktop supplements which endeavor to derive contextual associations or support their discovery.

UMEA [8] is an application that tracks activity and the objects of those activities, and creates a History log organized according to projects. Metadata describes the context in which the work is being carried out, which can then be used for retrieving contexts. UMEA, however, requires users to set up projects and then manually to switch between them in the UMEA interface. If a document is opened during a particular project context, then the document is associated with that project by UMEA. This can lead to mode errors, where the user forgets to switch project contexts before performing an action. This leads to the action or object being mistakenly classified, for example as belonging to the "workshop" project rather than the "funding proposal" project., UMEA therefore allows manual reclassification. Like Timescape, the interface allows different views, including a calendar view. It also allows the launching of PIM applications such as. Sticky notes, to do lists, and emails to project related contacts.

Milestones [13], is a visualisation for Stuff I've Seen [6] – a Microsoft research desktop search tool. It uses events and images from the user's wider context (such as headlines from world or local news, digital photographs the user took and stored on the computer at that time, etc) to act as landmarks and cue and orient the user in a timeline view (of search results). Episodic memory is therefore used to cue the user's recall of context, and was found to speed retrieval of desired items from search results compared to a view with no landmarks. The tool does not currently support user-authored annotation of the things shown for cuing context.

OnCue [4] rather than watching file I/O, monitors the clipboard in order to provide associated available services from postal code look ups to historgram generation from table data. OnCue is inspirational in the kinds of context-aware services it provides, and with which we would wish to supplement any contextual association of information.

Virtual Notebooks

Virtual Notebooks, like their physical cousins, support note taking and artefact pasting. They also provide additional digital features which enhance their data collection value. Some exemplars are Tinderbox (eastgatesystems.com) NoteTaker (aquaminds.com) and NoteBook (circusponies.com) which support direct entry of information, such as pasting in screen shots or web information, making notes or outlines, and publishing contents of pages or whole notebooks to the web. Most can output to XML and provide indexing for rapid searches. Tinderbox adds the interesting feature of providng agentware to data mine collections of information in order to find new possible associations in the data not previously noticed. We are strongly interested in the features which these notebooks provide for freely associating and cataloging multiple types of media. These applications are designed, however, for user-determined addition of content to the books. Our approach will be an effort to generate much of the content by the discovery of implicit contexts, supplemented by opportunities to add, subtract or annotate content manually.

Semantic Web Frameworks

The Semantic Web utilizes metadata that is represented in triples of subject-predicate-object. This simple structure can then be associated with ontologies representing classes of entities. The power of this ontology-informed approach to metadata means that we can use inference to derive new knowledge not explicitly stated in the data. Two different files which say nothing in their content that would relate them may still be inferred to be related based on some other association apparent in the metadata, as mediated through an ontology. It is this power of association that we think can be most valuable in helping to connect a scientists' local information with global contexts. As it stands, the eScience project in its utilization of the Grid (or what the NSF in the States refers to as "cyberinfrastructure) has been developing technologies which support the Semantic Web as a communication layer for eScience Grid applications. In an earlier project with synthetic chemists, we were able to capture their experiments in plain English and translate these into Semantic Web parlance for concurrent publication of results to the Grid [7]. The use of an

ontology for mapping the data meant that other services could use that ontology for interpreting these results against their own, and thus know how to process this data for their own requirements. It is because of this local/global flexibility for data reuse that we are interested in supporting a semantic web layer as a way to mediate context histories.

Certain frameworks already exist which we are exploring for adaptation in the current project. Haystack from MIT [11], at two years old, is the most mature. It provides a framework for developing Semantic Web applications. Its core demonstrator has been a personal information management system. Like virtual notebooks, it relies on the manual capture of information, but its use of a semantic back end, through ontologies, allows inferencing over data. In this way, making a plane booking will result in a calendar being updated with new location information for the dates away. To date their have been known issues with speed in applying to real world data, and predictable resistance to using one monolithic tool rather than being able to use one's own communication and scheduling tools. Recently, Haystack has been refining its framework and working on speed so we looking forward to exploring this further.

UTOPIA is another eScience Semantic Web project which can monitor activities in a defined virtual disk/work environment [10]. While such monitoring is potentially ideal for deriving context histories and translating them to a Semantic Web layer, it requires scientists to use a network disk mounted on their desktop. Files are saved to this virtual disk. As we will look at later, there is considerable apprehension in the community to having data stored on a remote device rather than first and foremost on one's own hard drive. Utopia is also as yet an early technology, not yet released as a framework. The Utopia group, however, is keen to have feedback from the interaction community in order to understand better what services/interactions in needs to support.

One of the chief concerns relayed to us from the bioinformatics community is the need to have flexible visualizations. mSpace (www.mspace.fm) is an interaction model currently implemented on Semantic Web protocols. The model supports user-determined arrangement of an information space in order to support exploration of relationships of the data from multiple perspectives. We are looking at adapting the mSpace software framework to provide local visualizations of the relations in the information.

APPROACH

As can be seen, there are already a variety of tools we can draw on for supporting the types of transparent interactions we wish to explore in utilizing context histories. We are not committed to any particular tool or framework, nor do we need to be, since our main goal is to explore the interactions we may be able to support in using context histories.

In keeping with the EScience lean towards Semantic Web technologies we do wish to create a semantic layer that can translate activities into the appropriate formats for semantic web service interaction.

In terms of interaction, while we want to be able to generate notebook type reports, it is clear from our early ethnography that requiring scientists to manage digital notebooks while carrying out digital experiments has a higher cost in terms of required steps than using a paper lab book. This kind of forced divided attention between file management and experimental activities is counter productive. Therefore we will investigate leveraging the type of transparent capture of file I/O activities demonstrated in Timescape and Presto. We will not be replacing the desktop, though, but will want the kind of project-sensitive associations found in UMEA, but without the required manual context switching. While we can leverage certain cues for context discovery -- a search in a gene database is likely part of an experiment; a search for a bike is less likely to be part – other kinds of cues, such as time, are more problematic. What is unimportant today may prove important tomorrow. Therefore tracking versioning on digital artifacts in a way similar to Timescape may be significant for recovery of context.

Risks of Contexts: Concerns for Design

In bioinformatics, privacy/security of data is a critical concern: any contextual history will almost always be reflecting traces of privileged data. This engagement with privileged data also relates to notions of perceived confidence in any deliverable system. Currently privileged bioinformatics data and related material is kept locally by individual scientists on computers they control. Solutions like Utopia which can only trace file I/O by using networked services rather than locally controlled machines are viewed with suspicion. Likewise, companies sometimes provide privileged data to bioinformatics scientists. These stakeholders will also need to be convinced that their data will be secure. Exposing the context of what scientists are doing with this data could be considered a risk. We will therefore examine current sharing practices, contexts and investigate our users' desired levels of data confidentiality, integrity and availability in order to design lightweight authorization models and authentication protocols. This may include expanding our design space from users to stakeholders, so that it includes data donors as well as data recipients. Where privacy is a concern, deploying encryption is initially compelling. However, it is a powerful technology that is often poorly implemented, causing the best a false sense of security, and at worst severe risks to data availability. Finally, we may need to investigate the effect of transparency of security solutions on bioinformatics users' trust in and desire to use our system - to tread the fine line between security's visibility and intrusiveness.

ACKNOWLEDGMENTS

Thank you to our reviewers. This work is supported by the myTea eScience Best Practice Project, mytea.ecs.soton.ac.uk, funded by the EPSRC, UK.

REFERENCES

1. Aboud, Gregory. Common Features of Ubicomp Applications. ICSE99.
2. Jakob Bardram Olav W. Bertelsen Supporting the Development of Transparent Interaction Lecture Notes In Computer Science; Selected papers from the 5th International Conference on Human-Computer Interaction. 1015, (1995):79-90.
3. De Roure, D., Jennings, N., Shadbolt, N. Research Agenda for the Semantic Grid: A Future *e*Science Infrastructure, in *Grid Computing: Making the global infrastructure a reality*, Berman, F., Fox, G., and Hey, T. (eds), Wiley Europe, 2003, 437-470.
4. A. Dix, R. Beale and A. Wood (2000). Architectures to make Simple Visualisations using Simple Systems. Proceedings of Advanced Visual Interfaces - AVI2000, ACM Press, pp. 51-60.
5. Dourish, P., Edwards, W. K., LaMarca, A., & Salisbury, M. (1999). Presto: An experimental architecture for fluid interactive document spaces. ACM Transactions on Computer-Human Interaction.
6. Dumais, S., Cutrell, E., Cadiz, J. J., Jancke, G., Sarin, R., & Robbins, D. C. (2003). Human interaction: Stuff i've seen: A system for personal information retrieval and re-use. Paper presented at the SIGIR '03: 26th annual international ACM SIGIR conference on Research and development in information retrieval, Toronto, Canada.
7. Hughes, G., Mills, H., de Roure, D., Frey, J., Moreau, L., schraefel, m. c., Smith, G. and Zaluska, E. (2004) The semantic smart laboratory: a system for supporting the chemical eScientist. Organic and Biomolecular Chemistry 2:pp. 1-10.
8. Kaptelinin, V. (2003). Umea: Translating interaction histories into project contexts. Paper presented at the CHI '03, Ft. Lauderdale, Florida, USA.
9. Kurtenbach, G., Buxton, W. Issues in combining marking and direct manipulation techniques, In Proc. of UIST, 1991, pp.137-144
10. S. Pettifer, J. R. Sinnott, and T. K. Attwood. UTOPIA: user friendly tools for operating informatics applications. Comparative and Functional Genomics, 5:56-60, January 2004.
11. Dennis Quan, David Huynh, and David R. Karger. Haystack: A Platform for Authoring End User Semantic Web Applications in ISWC 2003.
12. Rekimoto, J. (1999). Time-machine computing: A time-centric approach for the information environment. Paper presented at the 12th annual ACM symposium on User interface software and technology, Asheville, North Carolina.
13. Ringel, M., Cutrell, E., Dumais, S., & Horvitz, E. (2003). Milestones in time: The value of landmarks in retrieving information from personal stores. Paper presented at the Interact 2003, Zurich.
14. schraefel, m. c., Hughes, G., Mills, H., Smith, G., Payne, T. and Frey, J. (2004) Breaking the Book: Translating the Chemistry Lab Book into a Pervasive Computing Lab Environment. In Proceedings of CHI 2004, Vienna, Austria
15. schraefel, m. c., Wigdor, D., Zhu, Y. and Modjeska, D. (2002) Hunter gatherer: within-web-page collection making. In Proceedings of CHI '02 extended abstracts on Human factors in computer systems, pages pp. 826-827.
16. Robert Stevens, Hannah J. Tipney, Chris Wroe, Tom Oinn, Martin Senger, Phil Lord, Carole Goble, Andy Brass, and May Tassabehji. Exploring Williams-Beuren Syndrome Using myGrid. Bioinformatics, 20:i303-i310, 2004.
17. Jun Zhao, Chris Wroe, Carole Goble, Robert Stevens, Dennis Quan, and Mark Greenwood. Using semantic web technologies for representing e-science provenance. In Proc. of the Third International Semantic Web Conference, Lecture Notes in Computer Science, pages 92 - 106, Hiroshima, Japan, 2004. Springer.

Author Biographies

mc schraefel is a Senior Lecturer in the Intelligence, Agents and Multimedia Group, Electronics and Computer Science, University of Southampton. Prior to joining Southampton, schraefel completed a post doc at AT&T Labs Online Platform Research Group, Florham Park, and was an Assistant Professor in Computer Science at the University of Toronto, Canada. Her main area of research is interaction design for information systems and ubiquitous computing environments.

Sacha Brostoff is an Ergononist specialising in computer security, with an interest in e-Science. He gained his PhD at University College London studying the usability of computer password systems in large organisations.

Ray Cooke has had industrial experience working as an ASIC design engineer prior to receiving his B.Sc. in Computer Science with Advanced Networks and Distributed Systems from Southampton University in 2004. He is now working as a Research Assistant in eScience information systems and usability for Southampton University.

Andrew Gibson holds a B.Sc. in Biochemistry from UMIST, an MRes from the University of Leeds and a PhD in Bioinformatics from the University of Manchester. He is currently a research associate involved in bringing e-science practices into the bioinformatics domain.

Robert Stevens has qualifications in biochemistry, computational biology and computer science, where he originally specialised in HCI. His bioinformatics research centres around the use of computational representations of knowledge that facilitate analysis of biological data. the performing of science in silico draws together both the computational analysis and human aspects of the scientific process into his current research interests.

Proceedings of ECHISE'05, held in Conjunction with PERVASIVE'05, May 11, 2005, Munich, Germany.

Using Context History for Data Collection in the Home

Daniel H. Wilson
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
dwilson@cs.cmu.edu

Danny Wyatt
Dept. of Computer Science & Engineering
University of Washington
Seattle, WA 98195
danny@cs.washington.edu

Matthai Philipose
Intel Research Seattle
1100 NE 45th St., 6th Floor
Seattle, WA 98105
matthai.philipose@intel.com

ABSTRACT

Practical in-home health monitoring technology depends upon accurate activity inference algorithms, which in turn often rely upon labeled examples of activity for training. In this position paper, we describe a technique called the context-aware recognition survey (CARS) – a game-like computer program in which users attempt to correctly guess which activity is happening after seeing a series of symbolic images that represent sensor values generated during the activity. We describe our own implementation of the CARS, introduce preliminary results, and discuss the first steps toward a completely unsupervised system.

INTRODUCTION

Pervasive computing applications implicitly gather *context history* as they collect and store sensor data over time. In this position paper, we describe the context-aware recognition survey (CARS), which employs context history to help users label anonymous activity episodes. User-labeled examples of activity are valuable because they can 1) improve pervasive computing design decisions and 2) be used to train machine learning algorithms that recognize activities.

Drawing on recent research in practical home monitoring systems, game-based image-labeling techniques, and data visualization techniques [2,6,7], we designed a game-like multiple choice test that displays low-level sensor readings as colorful symbols and descriptive text. Users answer the questions with the goal of correctly labeling the activity being depicted. We report a study in which users (N=10) performed a subset of tasks in an instrumented environment and completed a context-aware recognition survey approximately one week later.

RELATED WORK

Several standard classes of methods exist for collecting data about daily activities, including one-on-one or group interviews, direct observation, self report recall surveys, time diaries, and the experience sampling method (ESM) [1, 4]. While direct observation is often reliable, it is prohibitively time-consuming. In interviews and recall surveys, users often have trouble remembering activities and may censor what they do report. Cognitively enhanced recall surveys mitigate forgetfulness by using cues such as photo snapshots. Time diaries also reduce recall and selective reporting bias, but require a commitment from the user to carry around (and use) the diary. Experience sampling uses a prompting mechanism (e.g., a beep) to periodically ask the user for a self-report. These prompts may interrupt activities and must be carefully delivered in order to avoid annoying the user [4]. All of these methods require the participation of the person who performed the activity and others may require outside help as well (e.g., interviewers).

CONTEXT AWARE RECOGNITION SURVEY

The key idea of the context-aware recognition survey is to use contextual information collected by ubiquitous sensors to provide an augmented recall survey that can be performed by anyone at any time, regardless of who performed the activity or how the sensors were configured. The technique consists of the following steps: 1) sensor readings are collected over time and stored, 2) sensor readings are automatically segmented by activity into episodes (called *episode recovery*), 3) episodes are converted into a series of generic, highly descriptive images, and 4) episodes are labeled by users in a game-like computer-based recognition survey. Afterwards, the labeled episodes may be used to train machine learning algorithms or to improve design decisions for pervasive computing applications.

Initial Study

We performed an experiment in which we designed, implemented, and tested a context-aware recognition survey. We now briefly describe the study.

Subjects. We recruited 10 adult volunteers from the university and from the community. Subjects ranged in age from 25 to 32 years, and the sample was 50% female and 50% male. Subject background varied, ranging from librarians to engineers.

Instrumented environment. This study occurred in the author's home. A kitchen and bathroom were instrumented with two types of anonymous, binary sensors: magnetic contact switches and pressure mats. Contact switches were placed on doors and drawers (e.g., refrigerator door, cabinet door, kitchen drawers). Pressure mats were placed in front of important areas (e.g., in front of the sink). Sensors were polled every second and values were stored in a mySQL database.

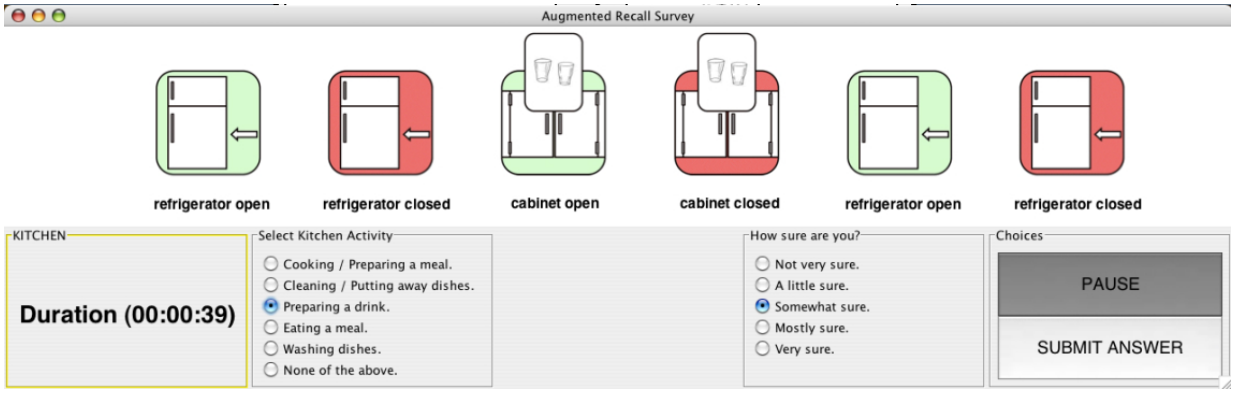


Figure 1. Screenshot of program.

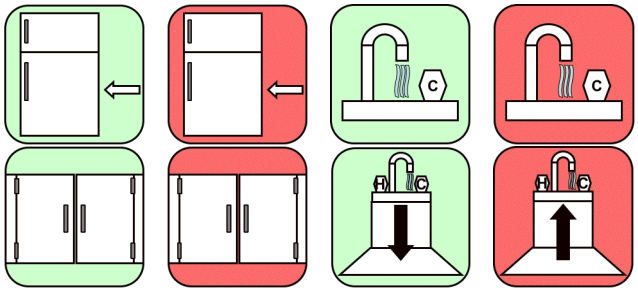


Figure 2. From left to right, top to bottom: (a) Refrigerator open, (b) refrigerator close, (c) cold water on, (d) cold water off, (e) cabinet open, (f) cabinet closed, (g) stand near sink, (h) leave sink.

Activity recording. Subjects were instructed to choose and perform a subset of several kitchen tasks. The kitchen tasks were: prepare a cold drink, prepare either a sandwich, a fried egg, or a microwave pizza, eat the meal, wash dishes and put them away, and throw away any trash. During the bathroom portion, subjects were given a toothbrush and were instructed to brush their teeth and then perform two of three tasks: washing their face, washing their hands, and combing their hair. An observer time-stamped the start and end points of each activity using a laptop computer. Subjects participated one at a time.

Context-Aware Recognition Survey. We presented our computer-based recognition survey as a "game" in which the goal was to correctly guess which activities were happening given only the sensor readings collected from the kitchen and bathroom environments. The contextual information gathered by the sensors was hand-segmented into episodes and converted into a series of images via the Narrator program [7].

See Figure 1 for a screenshot of the computer program. Each episode consisted of a series of scrolling images that had red or green backgrounds, depending on whether that object was turned on or off (see Figure 2). The word "kitchen" or "bathroom" was presented with each episode to indicate the location of the episode. The only timing information included was the total duration of the episode. Subjects were able to pause the scrolling pictures, but were not able to replay an episode. After viewing an episode, subjects were asked to select from a multiple choice list of every possible kitchen or bathroom activity (depending on which room the activity occurred in) plus a "None of the Above" answer. Subjects were also asked to rate how confident they were about their choice on a scale of one to five.

Subjects were administered the CARS on a laptop computer a mean of 5 days following the activity recording. Each subject was presented with two sets of 12 activity episodes, which we call the self set and the other set. The self set contained 8 episodes from the subjects own activities and 4 counterfeit episodes which did not correspond to any activity. The other set contained 8 episodes of someone else's activities and 4 counterfeit episodes. Subjects were informed of which sets were self or other. The survey administration was counterbalanced, with half of the subjects presented the self set first, and the other half with the other set first.

Results

Here, we discuss selected results of our study. See [9] for a more detailed discussion of results.

- Subjects successfully identified 82% of the 24 total episodes (M = 19.60, SD = 3.47). This indicates that **context history is useful for data collection in the home.** Indeed, subjects were able to successfully label most activities with confidence: on the Likert scale of 1-5 (1=Not Sure and 5=Very Sure), subjects reported being Mostly Sure (M = 3.96, SD = 1.03) across all of the episodes. Futhermore, user confidence ratings were significantly related to whether the episode was actually rated correctly, with a significant difference between mean confidence level on correct (M = 3.03, SD = 1.03) vs. incorrect (M = 2.61, SD = 1.06) selections, $t(238) = 2.39, p < .01$.

- Overall, **subjects were equally good at labeling their own or other people's activities**. Ignoring counterfeit episodes, performance on the self section (M = 7.10, SD = 1.29) and the other section (M = 7.10, SD = .99) was identical, with subjects correctly identifying 89% of the 8 possible episodes.

Figure 3. The iBracelet, a wearable RFID reader.

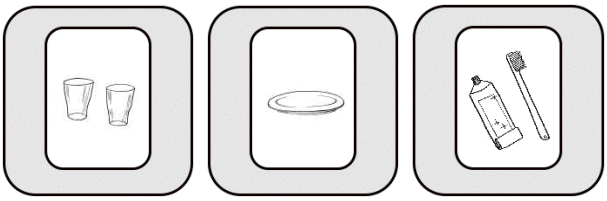


Figure 4. From left to right: (a) Cups, (b) plate, (c) toothbrush & toothpaste.

- The number of days between activity performance and activity recall ranged from 2 to 7 (M = 5.00, SD = 1.63) and was not significantly correlated with total performance scores, $r(8) = .27, p = .44$. This indicates that **context history may help mitigate recall bias.**

- We found that the order of test administration (self then other, or vice versa) impacted performance on the identification of counterfeit episodes. Subjects who completed the self section first were significantly better at detecting fake episodes in the other section ($t(8) = 2.36, p < .05$), indicating that **as subjects gained more practice their performance improved.**

- Subjects reported that they enjoyed using the program, calling the symbols "cute," and "easy to understand." Subjects reported that the symbolic images were "pretty easy" to "very easy" to understand on a Likert scale of 1-5 (*M* = 4.70, *SD* = .48). Thus, we found that using **a scrolling set of symbolic images was a useful approach for displaying context history.**

CURRENT WORK

We identified two main weaknesses in our CARS implementation: 1) we used low-granularity sensors (e.g., contact switches), and 2) we depended on a human to hand-segment the data into episodes. In this section we describe our current solutions in these areas.

Higher Granularity Sensors

In our study, we found that our choice of simple sensors did not provide sufficient granularity for users to confidently label certain activities. For example, it was particularly difficult to tell the difference between washing hands and face. To remedy this situation, we have begun to integrate higher granularity RFID sensors, specifically the iBracelet [5].

Figure 3 illustrates the RFID infrastructure that we assume. On the left is a bracelet which has incorporated into it an antenna, battery, RFID reader and radio. On the right are day-to-day objects with RFID tags (battery-free stickers that currently cost 20-40 cents apiece) attached to them. The reader constantly scans for tags within a few inches. When the wearer of the bracelet handles a tagged object, the tag on the object modulates the signal from the reader to send back a unique 96-bit identifier (ID). The reader can then ship the tag ID wirelessly to a base computer which can map the IDs to object names. We currently assume that subjects or their caregivers will tag objects; we have tagged over a hundred objects in a real home in a few hours. The corresponding CARS symbols are images of the objects being manipulated. We assembled several dozen prototypical object-symbols using the image search function of the Google search engine. See Figure 4 for example symbols.

Automatic Episode Recovery

An attractive aspect of the context-aware recognition survey is the fact that it is completely unsupervised (aside from the user labeling step). In our previous study, however, we hand-segmented the stream of sensor readings generated by the user. In a first step towards automating this step, we conducted a small study that used HMMs bootstrapped with common sense information mined from the Internet. The key idea is to train rough HMM models with information "scraped" from instructional web pages, and then to use these models to identify the segments between activity episodes.

We conducted an experiment to test the usefulness of bootstrapped HMMs for automatic episode recovery. We used data from a previous study in which over 100 RFID tags were deployed in a real home. Objects as diverse as faucets and remote controls were tagged. We had 9 non-researcher subjects with a wearable RFID reader perform, in any order of their choice, 14 ADLs each from a possible set of 65; in practice they restricted themselves to 26 activities over a single 20 to 40 minute session. There were no interleaved activities and a written log was used to establish ground truth.

An HMM was trained on information gathered from the Internet. The datamining process used word appearances on "how to" websites to compute the probability that an object was used during each activity. From this mined information we assembled an HMM with one state for each activity, and a set of observations composed of the set of mined objects, pruned to include only those which we know are in our set of deployed tags. The observation probabilities were set to normalized values of the mined probabilities. We set the HMM's transition probabilities to reflect an expected number of observations (5) for each activity, as well as a uniform probability of switching to any other activity. See [5] for a thorough description of the datamining process.

Next, for each of the 9 sensor traces (one for each subject) we used the Viterbi algorithm to compute the most likely sequence of labels for each object (i.e., sensor reading). We then simply segmented the labeled trace into contiguous sequences of the same label. To measure accuracy of the segmentation we used the P_k metric [3]. The P_k metric is the probability that two observations at a distance of k from one

another are incorrectly segmented. As such, it can be thought of as the error rate for the segmentation and 1 - P_k can be thought of as the segmentation's accuracy. k is set to one half of the average segment length (3 in our case). The P_k score for our segmentation using only the mined parameters is 29.7, indicating that we should expect to be able to segment sensor traces in a completely unsupervised manner with higher than 70% accuracy. This indicates that **bootstrapped HMM models can potentially perform unsupervised episode recovery**.

EXPECTATIONS FOR THE WORKSHOP

Context history is a powerful source of information with many exciting applications. The ECHISE workshop provides the first author an opportunity to meet other researchers who are using similar technologies and approaching similar issues. Moreover, it offers a valuable opportunity to achieve consensus among other researchers as to problem areas and promising avenues of future research.

We are interested in determining how other researchers are using context history in terms of pervasive computing. Specifically, we are interested in sharing tips and techniques for using context history in the domain of automatic health monitoring – an increasingly important application of pervasive technology. How other researchers collect context history, what they choose to collect, and how they present it is of interest. Finally, we are particularly interested in learning how other researchers are dealing with privacy constraints.

CONCLUSION

In this position paper, we described current work with the context-aware recognition survey, an approach for labeling activities that uses contextual information collected by sensors. We presented results from a recent user study, indicating that such an approach can be effective. We discussed improvements being incorporated into the next generation our own CARS. Finally, we described what we hope to get out of the workshop.

AUTHOR BIOGRAPHIES

Daniel H. Wilson is a fourth year Ph.D. candidate in the Robotics Institute of Carnegie Mellon University, where he has received masters degrees in robotics and data mining. His research goal is to provide simultaneous tracking and activity recognition for multiple occupants in the home.

Danny Wyatt is a Ph.D. student in the Department of Computer Science & Engineering at the University of Washington. His research interests include sensing and modeling human behavior.

Matthai Philipose is a researcher at Intel Research Seattle. His primary areas of interest are programming languages and probabilistic reasoning. He is currently working on sensors, data modeling, and statistical reasoning techniques for recognizing human activities.

REFERENCES

1. L. F. Barrett and D. J. Barrett. An introduction to computerized experience sampling in psychology. *Social Science Computer Review*, 19(2):175-185, 2001.
2. C. Beckmann, S. Consolvo, and A. LaMarca. Some assembly required: Supporting end-user sensor installation in domestic ubiquitous computing environments. In *Proc. of UBICOMP 2004*, 2004.
3. D. Beeferman, A. Berger, and J. Lafferty. Statistical Models for Text Segmentation. Machine Learning. 34(1-3):177-210, 1999.
4. S. Intille, E. M. Tapia, J. Rondoni, J. Beaudin, C. Kukla, S. Agarwal, and L. Bao. Tools for studying behavior and technology in natural settings. In *Proc. of UBICOMP 2003*, 2003.
5. M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, H. Kautz, and D. Hahnel. Inferring activities from interactions with objects. IEEE Pervasive Computing Magazine 3(4):5057, 2004.
6. L. V. Ahn and L. Dabbish. Labeling images with a computer game. In *Proc. of CHI 2004*, pages 319-326, 2004.
7. D. H. Wilson and C. Atkeson. The narrator: A daily activity summarizer using simple sensors in an instrumented environment. In *Adjunct Proc. of UBICOMP 2003: Demonstrations*, pages 141-144, 2003.
8. D. H. Wilson. Simultaneous tracking and activity recognition (STAR) using many anonymous, binary sensors. *Ph.D. Thesis Proposal*, CMU, June 2004.
9. D. H. Wilson, A. C. Long, and C. Atkeson. A context-aware recognition survey for data collection using ubiquitous sensors in the home. In *Proceedings of CHI 2005: Late Breaking Results*, pages 1856-1857, 2005.

Situation Determination with Distributed Context Histories

Graham Thomson, Paddy Nixon, and Sotirios Terzis
Global and Pervasive Computing Group, Department of Computer and Information Sciences
University of Strathclyde, Glasgow, UK

ABSTRACT

Determining the situation within an environment is a key goal of smart environment research. A significant challenge in situation determination is reasoning about open-ended groups of people and devices that a smart environment may contain. Contemporary solutions are often tailored to the specific environment. In this position paper, we present a novel general situation determination framework, that by viewing people and tools as playing roles in a situation, can easily adapt recognition to incorporate the dynamic structure of a situation over time.

INTRODUCTION

Determining the situation within an environment is a key goal of smart environment research. It provides a natural pivot to which users and application programmers can associate behaviours, such that the computing machinery contained within the environment silently and automatically adapts to its inhabitants' behaviours, "invisibly enhancing the world" [1].

The approaches to situation determination offered by the state-of-the-art context-aware infrastructures [2, 3] experience the following drawbacks:

- An expert of the particular environment is required to specify the correlation of the available context information with the situations that occur. Reasoning is performed by large logic programs [3] or Bayesian networks [2], which must be manually constructed and maintained.
- As the amount of available context information and number of situations increases, it becomes increasingly difficult for an expert to decipher and specify correlations.
- The situation specifications will suffer from the subjective bias of the expert who programmed them.
- Recognition is limited to the fixed number of cases programmed by the expert for the local environment, and does not adapt well to the introduction of unrecognised people or devices in the environment, that is, when the structure of the situation is uncertain.

This paper presents a novel approach to situation determination that attempts to address these issues. There is no need of an environment expert, as situations are programmed by example by users themselves. Reasoning is based upon a general situation determination framework that can be applied in any smart environment. By viewing people and tools as playing roles in a situation, recognition can easily adapt to incorporate the dynamic structure of a situation over time.

To illustrate the kind of situations we wish to detect, listed below are examples of typical situations that occur within our department, along with a description of their characteristics.

Project meeting People that are members of a project are assembled in a meeting room.

PhD meeting A PhD student and their supervisor are talking in the supervisor's office.

Conversation Two or more people are talking in the same area.

Presentation An audience is assembled in a meeting room, a projector is running, and presentation software is running. The host introduces the speaker(s), the speaker(s) present, and then the speaker(s) answer questions posed by the audience.

Checking mail A person is working with mail reader, web browser, and document reader tools, with the mail reader tool being used most frequently.

Reading A person is alone at their desk, and the computer or any other tools in the desk area are idle.

Coffee break The time is around either 11 or 4 o'clock, and a group of people are assembled in the staff room, drinking coffee.

Party In the late afternoon or evening, a large group of people are gathered in a conference room, with music playing and a projector displaying photographs.

From considering the situations presented above in addition to other situations from domestic and social scenarios, we propose that a situation can be robustly characterised by the combination of four main aspects:

Figure 1. The situation tree.

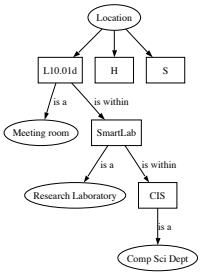


Figure 2. The Location subtree.

- The time at which the situation occurs, as well as its duration.
- Where the situation occurs, and the properties of that location.
- The attributes of the group of people that are present.
- The group of tools that are present, and the manner in which they are being used.

Furthermore, we propose that to accurately identify the elements of a situation that change over time, analysis of only the order and proportion of those elements is sufficient.

In the rest of this paper, we go on to describe the challenges faced when attempting to determine the situation, and present how we deal with these challenges in our approach to situation determination.

REPRESENTING THE SITUATION

An ontology based approach is used to represent the information that characterises a situation. Doing so permits matching at various levels of abstraction, information to be exchanged and interpreted correctly between multiple participants, and the information to be translated to different ontologies that may be used in other environments.

Figures 1 through 6 shows an example ontology describing a small presentation situation. The ontology is structured as a tree. At the root of the tree in Fig. 1, is the Situation class, which contains four other classes - Time, Location, People, and Tools.

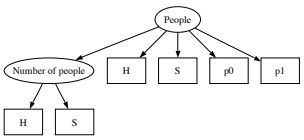


Figure 3. The People subtree.

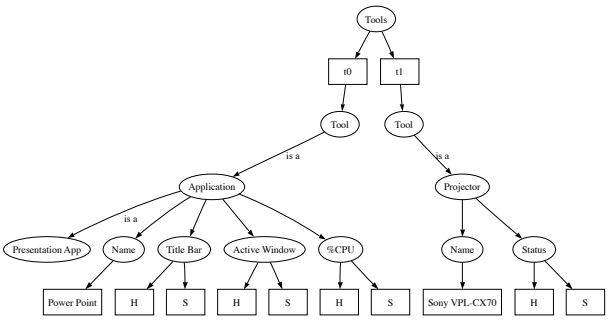


Figure 4. The Tools subtree.

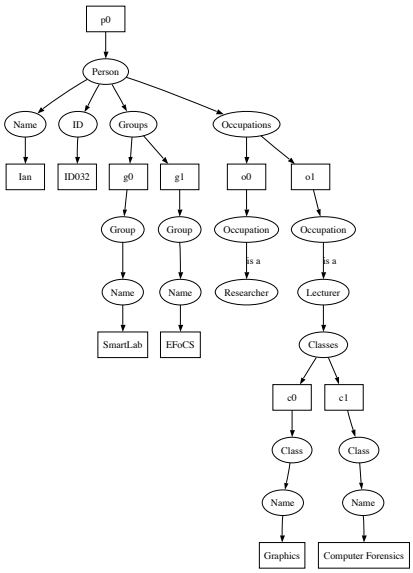


Figure 5. The p_0 subtree.

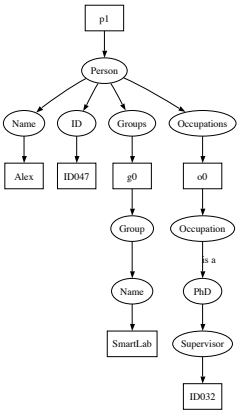


Figure 6. The p_1 subtree.

Time is the simplest aspect in the tree and includes values such as the time of day, day of week, date of month, etc. The tree is not shown in the interests of space.

To represent location, a symbolic, hierarchical model is used. That is, locations are referred to by name, and may be spatially contained in other locations. For example, Fig. 2 shows that 'SmartLab' is contained in 'CIS'. Each location is associated with a class that represents the type of location it is. For example, the location 'SmartLab' is a Research Laboratory.

As there may be a group of people present in a particular situation, each person is given a label, shown as p_0 and p_1 in Figs. 3, 5, and 6. These labels are utilised in matching, as we shall see later. From each person instance stems useful information that will help to identify a situation, such as their occupation, the groups they belong, and so on. For some situations such as a conversation, it is simply the number of people present, not their individual identities that are characteristic. A Number of People class is included to capture this explicitly.

Each tool that is involved with a situation is also given a label, shown as t_0 and t_1 in Fig. 4. Tools can include physical devices such as projector or a whiteboard, as well as computer applications. Computer devices are not regarded as tools themselves, as their use is characterised by the software applications that they host.

For each class in the ontology, its variables shall be either static or dynamic. If a variable is static, it is the value of the variable that characterises a situation. For example, in Fig. 5, 'Ian' is the value of the static variable 'Name'. If a variable is dynamic, it is the change in its value that characterises a situation. For example, in Fig. 4, the variable '%CPU' is dynamic. To capture the change in value, the sequence and histogram of values is stored for a dynamic variable. A sequence captures the order of the values, a histogram captures the proportion of the values. In the figures, dynamic variables can be identified by their Sequence (S) and Histogram (H) siblings.

Recent work has shown success in using dynamic Bayesian networks to classify low-level actions or activities of users, such as holding a telephone handset, or adjusting a thermostat [4, 5]. However, applying such techniques to recognising a situation, where the number of variables is very large and may change continuously (the structure is dynamic), and the situation's duration may be an hour or more, would be impractical. We suggest that such techniques are ideal for generating values of variables in the situation tree. For example, our current implementation uses a hidden Markov model to infer the current location.

SITUATION DETERMINATION

In our approach, situations are captured by the users themselves. We envisage a user running a client that requests a label for the situation, as well as the start and end times. During this period, the situation tree for the current environment (the context history) is recorded. Currently, we define the environment to be the room the user is in.

When determining the situation, we are comparing a situation tree that reflects the current state of the environment to a situation tree that has been previously captured and labelled. We shall refer to these as an example situation and the current situation respectively.

For each class defined in the ontology, there exists a comparison function that returns the similarity of two instances of that class. In this way, the similarity instances can be given the most appropriate score according to their class. For example, as the string 'Alex' is an instance of the class Name, it would have a greater similarity when compared to 'Alexander' than to 'Alan'.

It is the open-ended number of entities that makes situation determination challenging, specifically matching groups of entities. When a situation contains an open-ended group of objects, its structure is dynamic. As a situation is strongly characterised by the groups of people and tools it contains, an efficient method to compare groups is required. Consider the presentation application tool that is part of the Tools group in Figure 1. It is PowerPoint, and let's assume its dynamic properties show it is running lightly continuously. In an example situation it may be OpenOffice Impress that is running lightly continuously. What we are interested in is finding the tool that is playing the *role* of a presentation application running lightly continuously. Similarly in a PhD meeting situation, within the group of people present a person plays the role of a supervisor, while another plays the role of a student. Any group can be viewed as a set of roles. The problem of matching a group is then finding which members of a group from the current situation best fit the roles defined by the group in an example situation.

In Figs. 3, 5, and 6 the two instances of Person are labelled p_0 and p_1. These labels identify the role that that person plays. To compare a group, we have to use a more sophisticated compare function than that described previously. To illustrate, we shall consider the case of matching two People groups. Let ex.P be the People group from an example, and c.P be the People group from the current situation. The algorithm for matching groups is then:

1. For each ep_i in $ex.P$ and cp_j in $c.P$, compute the similarity of ep_i and cp_j, $sim(ep_i, cp_j)$.

2. Construct all possible mappings from the elements of $c.P$ to $ex.P$ where either every element of $ex.P$ is mapped to by a single, distinct element from $c.P$, or every element of $c.P$ maps to a single, distinct element in $ex.P$.

3. Calculate the score for each mapping by taking the average of the $sim(ep_i, cp_j)$ scores of each map within it.

4. The mapping which has the highest score gives the score for the group as a whole, as well as the optimal mapping of roles.

When comparing large groups, step 2 may become prohibitive. In cases where each member of a group contains only static properties, the computational cost can be reduced. To achieve this, each group member is given a unique id. When two group members are compared, their unique ids are compared first. If the unique ids match, we can shortcut the rest of the comparison of the two members as we know we have an exact match.

Members identified by their unique id can be confidently assigned to a particular role. Then, it is only the group members whose unique id is not recognised that must be assigned, substantially reducing the number of mappings that must be constructed.

Matching 'in vivo'

When an example situation is captured, the information in the situation tree is recorded over a length of time. When situation matching is performed, it must be done *as the situation is unfolding*. If only a single situation tree were used to capture an entire situation, we could only accurately match it at the end of the situation. Therefore, situation trees are captured periodically throughout the duration of a situation. The sequences and histograms of dynamic variables store the changes in value of the variable from the beginning of the situation up to the end of the period.

Distributed matching of situation fragments

So far we have looked at the situation tree as a whole. The information contained within the tree shall come from several different sources. It is therefore undesirable, and unnecessary, to have to collate the information in one place to perform matching.

In our approach, each person and tool has a corresponding software agent. The agent observes the information in, and performs matching on, the fragment of the situation tree that represents the person or tool. No single agent has a view of the entire situation tree, it can only see its own fragment. For example, in Fig. 5, the agent representing Person 'Ian' would see only the subtree starting at p_0, and in Fig. 4 the agent representing the Power Point instance, would see only the subtree starting at t_0. There is also an agent that represents the current environment, which we refer to as the situation server (SS).

When a tool or person enters a new location, its agent alerts the SS for that location to its representative's presence. The communication links between the SS and each agent in the environment form a star topology. Each agent stores locally applicable fragments of example situations. The agent compares the current situation fragment to each example fragment, making a list of all (fragment label, score) pairs. This list is then sent to the SS.

When the SS has received a list from each agent, it must combine these fragment scores into a score for that situation tree as a whole. The list from a Person agent will include the score of the agent's representative against each Person role in a situation, likewise for a list from a Tool agent. Based on these lists, the SS executes the group score / role assignment algorithm for the People and Tools groups. The SS then computes the scores for Location and Time and combines these to produce the total scores for each situation. The SS then sends a message back to each agent in the environment, indicating the highest scoring situation.

FUTURE WORK

In this paper it has been assumed that people in the same location are in the same situation. This will not always be the case. For example, in an open plan work area, some people may be involved in a conversation, while others may be working at their desk. Recognition could be extended to differentiate between these. Furthermore, some situations such as 'Journey home' will be characterised by a sequence of locations. In this case it may be inappropriate for a Person agent to attempt collaborative determination at each location.

As the number of situation examples increases, each agent will have a greater number of fragments to match. After an example situation is captured, a clustering phase could be introduced to reduce the total number of situation fragments.

In some situations, such as outdoor situations, it may not be possible to host a dedicated SS. In such cases, the agents involved in a situation would participate in an election protocol to identity the most suitable agent to act as the SS.

We are currently experimenting with our approach on simulated data, for which the initial results look promising. We are also evaluating different approaches to combining examples of the same situation, as well as opportunities for unsupervised learning. We are also working on the development of the necessary matching agents, which will provide a prototype system allowing us to experiment with real-time situation determination.

REFERENCES

1. M. Weiser. The computer for the 21st century. *Scientific American*, pages 94–104, September 1991.
2. A. Ranganathan, J. Al-Muhtadi, and R. H. Campbell. Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, 3(2):62–70, 2004.
3. H. Chen, T. Finin, and A. Joshi. A context broker for building smart meeting rooms. In *Proceedings of the Knowledge Representation and Ontology for Autonomous Systems Symposium, 2004 AAAI Spring Symposium*. AAAI, March 2004.
4. Matthai Philipose, Kenneth P. Fishkin, Mike Perkowitz, Donald J Patterson Dirk Hahnel, Dieter Fox, and Henry Kautz. Inferring Activities from Interactions with Objects. In *IEEE Pervasive Computing: Mobile and Ubiquitous Systems*, volume 3, pages 50–57. IEEE, 2004.
5. Kenneth P. Fishkin, Bing Jiang, Matthai Philipose, and Sumit Roy. I sense a disturbance in the force:

Unobtrusive detection of interactions with rfid-tagged objects. In *Ubicomp*, pages 268–282, 2004.

Author Biographies

Graham Thomson is a PhD student in the Department of Computer and Information Sciences at the University of Strathclyde. He received his BSc in Software Engineering from the University of Strathclyde. Contact him at the Univ. of Strathclyde, Dept. of Computer and Information Science, Livingstone Tower, 26, Richmond Street, G1 1XH Glasgow, Scotland; Graham.Thomson@cis.strath.ac.uk.

Sotirios Terzis is a lecturer in the Department of Computer and Information Sciences at the University of Strathclyde. He received his PhD in the Computer Science Department at Trinity College Dublin. His research interests include context-awareness and trust management in pervasive computing systems. He received his BSc and MSc with a specialization in distributed systems from the University of Crete. He is a member of the ACM, the IEEE Computer Society, and the British Computer Society. Contact him at the Univ. of Strathclyde, Dept. of Computer and Information Science, Livingstone Tower, 26, Richmond Street, G1 1XH Glasgow, Scotland; Sotirios.Terzis@cis.strath.ac.uk.

Paddy Nixon is professor of computer science at the University of Strathclyde where he leads the Global and Pervasive Computing Group. He received his BS and PhD in computer science from Liverpool University and his MA from Trinity College Dublin. Contact him at the Univ. of Strathclyde, Dept. of Computer and Information Science, Livingstone Tower, 26, Richmond Street, G1 1XH Glasgow, Scotland; paddy@cis.strath.ac.uk.

Immune Inspired Context Memory

Philipp H. Mohr
Computing Laboratory
University of Kent
Canterbury, UK
phm4@kent.ac.uk

Jon Timmis
Computing Laboratory
University of Kent
Canterbury, UK
jt6@kent.ac.uk

Nick Ryan
Computing Laboratory
University of Kent
Canterbury, UK
nsr@kent.ac.uk

ABSTRACT
This paper presents a context processing system, which stores context in an appropriate data structure and can provide a selective context history to a range of applications. Artificial Immune System algorithms are used to achieve data reduction, continuous online learning and forgetting of obsolete context.

Author Keywords
context history, context memory, artificial immune systems

INTRODUCTION

In order to set the scene for the content of the paper we first clarify our working dependencies:

> *Context* is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves [1].
>
> *Context history* is the total collection of recorded past context.
>
> *Context memory* is a mechanism for retaining and recalling interesting and relevant past experience.

The objective of our research is to build a context processing system which only stores *relevant* context in a context memory. The stored context is used to provide context-aware applications with a selective context history, or context memory; we believe such a system should fulfil the following requirements:

- *Minimal amount of storage:* Storing all context is intractable [7, 2], therefore the amount of stored context data needs to be kept to a minimum; by removing duplicate context data and 'forgetting' potentially obsolete context data we construct a context memory (i.e., a *selective* context history).
- *Layered design:* The system should be designed with a layered approach (see Figure 1) which provides services and information to a number of applications.
- *Episodic memory:* Relationships between consecutive events need to be highlighted [6].
- *Context data should be smoothed:* When the users' behaviour changes, the systems' perception of the users' common behaviour should change gradually, as a sudden change is not desirable [11]. The system should also not be affected by noisy context data.
- *Ubiquitous environment:* The system needs to be made available on small, portable, resource-constrained devices and it needs to work in a range of networking environments with the real possibility that it must spend a proportion of time working with no connectivity.
- *Every day environments:* In order for the system to diffuse into every day environments, the user should not be required to be an active part of the system.

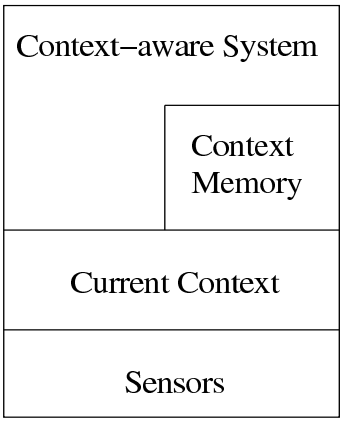


Figure 1. Layers.

Our proposal focuses on context data of an individual person. In order to create such a context memory we capitalise on techniques developed in the field of Artificial Immune Systems, as we believe they fulfil the

above requirements. A widely accepted definition of an Artificial Immune System (AIS) is:

> Artificial immune systems are adaptive systems, inspired by theoretical immunology and observed immune functions, principles and models, which are applied to problem solving [3].

For a system to qualify as an AIS it is required to have a minimum level of immunological inspiration incorporated, such as a model to perform pattern matching. AISs are often associated with virus detection, but their strengths are further reaching: they can perform pattern recognition, data compression, supervised and unsupervised learning, and be used to construct specialised memory structures. Calling something immunological does not make it an AIS. For a detailed introduction to AIS see [3] and [4].

In our context processing system we make use of AIS memory mechanisms, in particular Artificial Recognition Balls (ARBs) [10] which enable us to perform data reduction. A detailed explanation of ARBs is given in Section ; other AIS techniques relevant to our work are discussed in a previous paper [9].

Section describes our context processing system and Section presents our conclusions.

CONTEXT AWARE IMMUNE SYSTEM

In order to explain our Context Aware Immune System (CAIS) we define the *representation* of the components, the *affinity measures* which are used to quantify the interactions of the elements, and the *algorithm* which governs the behaviour of CAIS.

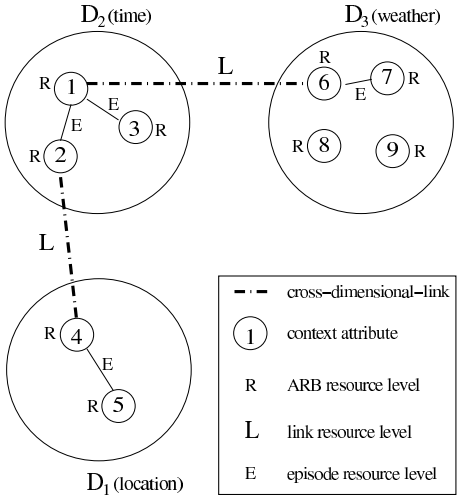


Figure 2. Data structure.

Representation:

The input to and output from CAIS is context represented by an attribute vector, $\langle a_1, a_2, ..., a_n \rangle$, which contains attributes (which can appear in an arbitrary order) along with their attribute identifier; for example:

```
⟨ Location.Building = Library,
  Wlan.MacAddress = 0A:40:C3:8D:00:32,
  Time.Hour = 18:30,
  Activity = Meeting ⟩
```

Where `Location.Building`, `Wlan.MacAddress`, `Time.Hour`, and `Activity` are attribute identifiers, and `Library`, `0A:40:C3:8D:00:32`, `18:30`, and `Meeting` are their respective values.

In order to store the attributes CAIS makes use of ARBs, which enable CAIS to perform data reduction since an ARB represents all elements in a region (in our case a 'region', and hence an ARB, is a hyper-sphere) eliminating the need for repetition. For example, a place of interest can be represented by an ARB instead of all individual GPS co-ordinates which fall within its radius; we associate an ARB with a resource level R, which increases when data points fall within its radius, and decreases by a constant decay. The radius r is a function of the resource level: $f(R) = r$.

The memory is an n-dimensional hierarchical network structure, where each dimension represents a different class of attributes. The attribute identifiers in our above example indicate to which dimension an attribute belongs, for example `Time.Hour` indicates that `18:30` belongs to the dimension 'time'. Figure 2 shows an example of the prototype data structure. The example consists of three dimensions and nine ARBs (drawn as small numbered circles). ARBs from different dimensions which appear in the same context are connected by cross-dimensional-links — this mechanism exploits the information present in the relations between context attributes. Furthermore, these links have a resource level L associated with them which reflects the likelihood that these two attributes occur in the same context. In our example ARB_4 and ARB_2 link dimensions D_1 and D_2, and ARB_1 and ARB_6 link dimensions D_2 and D_3. Furthermore, every dimension itself contains a network structure, for example dimension D_1 contains ARB_4 and ARB_5. In this structure, ARBs are connected with each other to highlight the relationship between consecutive events, which enables us to create an episodic memory. These links are associated with a resource level E to reflect the likelihood of certain events happening after each other.

Affinity Measures:

Every dimension has an affinity measure associated with it, which is used to determine the similarity between elements belonging to this dimension — for example, we use Euclidean distance to determine the simlilarity between GPS co-ordinates.

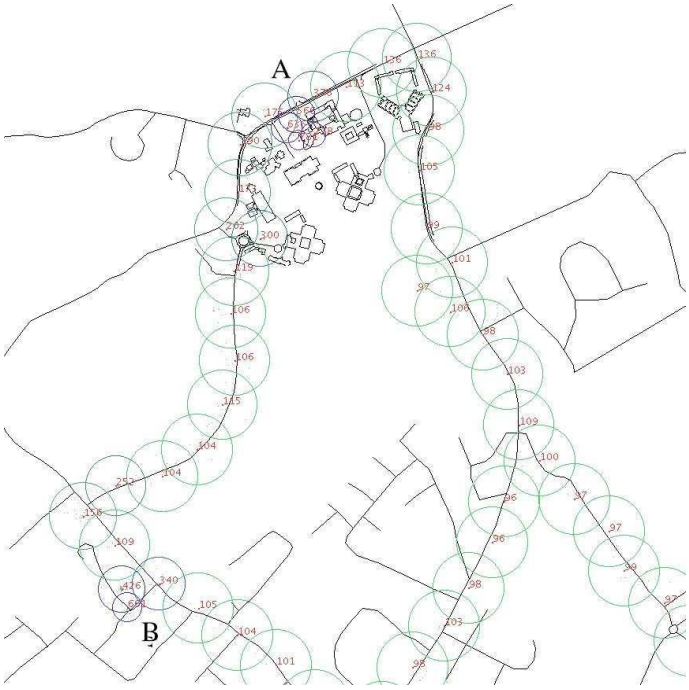


Figure 3. A snapshot of context memory, where the 65 circles represent ARBs. The smaller circles close to A and B show that these areas are stored with a high granularity.

Algorithm:
The algorithm used by CAIS is based on the principles of unsupervised and reinforcement learning and uses a combination of the representation and affinity measures described above; a suitable definition of unsupervised learning is:

> In unsupervised learning or clustering there is no explicit teacher, and the system forms clusters or "natural groupings" of the input patterns. "Natural" is always defined explicitly or implicitly in the clustering system itself [5].

Reinforcement learning can be defined as:

> Reinforcement learning addresses the question of how an autonomous agent that senses and acts in its environment can learn to choose optimal actions to achieve its goals. Each time the agent performs an action in its environment, a trainer may provide a reward or penalty to indicate the desirability of the resulting state [8].

Unsupervised learning allows us to construct a system which can cluster input data without any prior knowledge of the classes of the data. Reinforcement learning requires feedback from a trainer. However, an explicit trainer is not present in most context-aware systems, therefore an ARB in our system receives positive feedback when context attributes fall within its radius, and negative feedback is introduced by the notion of 'forgetting', which gradually decays all resource levels. For example, locations a user visits often have their rating reduced, but every visit increases the rating, which enables these locations to remain in the system. This also enables the system to reduce obsolete data, making the amount of data stored more manageable. Smoothing of context data is achieved by stimulation and decay of context attributes. The minimum amount of time an attribute remains in the system is controlled by the decay mechanism.

Having explained the individual components of CAIS, we now explain how they fit together. When CAIS receives an attribute vector the context processing algorithm selects the first attribute a and searches for the dimension it belongs to. In Figure 2 this might be attribute 1, which belongs to dimension D_2. Having found the dimension, it searches it for an ARB— using the appropriate affinity measure— which already covers the area of a; if such an ARB is found it is stimulated, which results in an increase in its resource level, R. If no such ARB exists, a new ARB is centered at a and initialised with a default resource level. The system then checks if the stimulated or newly created ARB matches the criteria for being part of an episode (e.g. if it is encountered shortly after the previously stimulated one); if it does, an existing link to the previous ARB is stimulated (e.g. the link between 4 and 5) or, if one does not exist, a new link is created with a default resource level, E. This process is repeated with all attributes in the vector. Next the system searches for existing links between attributes from this vector (e.g. cross-dimensional-link between 2 and 5). If links exist they are stimulated,

otherwise new ones are created with a default resource level, L. All ARBs and all links between them are subject to a decay mechanism to control the population.

If an outlier is chosen as the center of an ARB, this ARB will not receive enough stimulation and will die out, as the loss of resources due to the decay mechanism is higher than the resource gain. If the centre of an ARB is a sub-optimal center for the area which is covered by this ARB, the centre needs to be shifted. This is achieved by the stimulation and decay mechanisms. For example, if a person spends most of his time in his office, but the initial ARB covering the area of the building is created using the entrance as the centre of the ARB, the ARB should be shifted to a centre close to the persons' office. This shift happens when the initial ARB is stimulated to such a degree that it does not cover the area of the office any more, as the next location recorded outside this region, which is most likely to be close to the office, will form the centre of a new ARB. The new ARB will quickly gain resources as it is stimulated by the high activity in the area of the office, and the initial ARB will grow again as the activity in the area of the office now stimulates the new ARB. This process happens gradually over time and highlights the dynamic nature of CAIS.

A context-aware application may need access to current context, but also to context memory in order to identify recurring situations. As shown in Figure 1 a context-aware application may either query the underlying current context or the context memory [12]. If it requires current location data, for example to predict a route, it would obtain matching previous context data from the context memory.

Figure 3 presents preliminary results with a small subset of GPS co-ordinates. The data set consists of 3000 GPS data points which where collected by a single person over a period of 46 days. In the experiment we simulate this period by iterating through the data set. The 3000 data points are reduced by the algorithm to 65 (not all are visible in the figure), showing its data reduction capabilities. A large proportion of the data points where collected around position A and a slightly smaller proportion around position B, this is reflected in Figure 3 by small circles — representing a high data granularity. Less frequently visited areas and commonly traced routes have a lower data granularity and are reflected by larger circles.

CONCLUSION

Context memories have a great potential to improve context-aware systems, but constructing context memory structures or selective context histories is not straightforward. Such systems should be usable by multiple applications, therefore they need to be designed in such a way that they are generic enough to serve different applications. In an ideal world we want context processing systems to be able to generalise, be adaptable, and be able to compress or reduce the amount of data stored; immune inspired algorithms are good candidates, as they offer us mechanisms to fulfil all the requirements.

In this paper we propose an immune inspired context memory, which can adapt to a wide variety of user behaviour and environmental inputs. This is achieved by using immunological metaphors and immune inspired algorithms combining unsupervised and reinforcement learning.

REFERENCES

1. Abowd, G.D. and Dey, A.K. Towards a better understanding of context and context-awareness. http://www.cc.gatech.edu/fce/contexttoolkit/chiws/Dey.pdf
2. Aizawa, K., Hori, T., Kawasaki, S. and Ishikawa, T. Capture and efficient retrieval of life log. In *Proc. Pervasive Computing 2004 workshop on memory and sharing of experiences*, 15–20.
3. de Castro, L. and Timmis, J. Artifcial Immune Systems: A New Computational Approach. *Springer-Verlag, London. UK.* (2002).
4. de Castro, L. and Timmis, J. Artifcial Immune Systems as a Novel Soft Computing Paradigm. In *Soft Computing 7* (2003), 526–544.
5. Duda, R.O., Hart, P.E., Stork, D.G. Pattern Classifcation. *John Wiley & Sons Inc* (2000).
6. Hart, E., Ross, P., Webb, A. and Lawson, A. A Role for Immunology in "Next Generation" Robot Controllers In *Proc. Second International Conference on Artifcial Immune Systems*, LNCS Springer (2003), 46–56.
7. Harvel, L.D., Liu, L., Abowd, G.D., Lim, Y., Scheibe, C. and Chatham, C. Context cube: Flexible and effective manipulation of sensed context data. In *Proc. Pervasive Computing 2004*, LNCS Springer (2004), 51–68.
8. Mitchell, T.M. Machine Learning. *McGraw-Hill, New York* (1997).
9. Mohr, P.H., Ryan, N. and Timmis, J. Exploiting Immunological Properties for Ubiquitous Computing Systems. In *Proc. Third International Conference on Artifcial Immune Systems*, LNCS Springer (2004), 277–289.
10. Perelson, A.S. Immune network theory. In *Imm. Rev.* (1989), 5–36.
11. Petzold, J., Bagci, F., Trumler, W. and Ungerer, T. Global and local state context prediction. In *Artifcial Intelligence in Mobile System* (2003).
12. Ryan, N. Smart Environments for Cultural Heritage. In *Reading Historical Spatial Information from around the World, Proc. 24th International Research Symposium* (2005), 71–80.

Philipp H. Mohr received his BSc Honours in Computer Science from the University of Kent in 2003. He is currently a PhD student in the Pervasive Computing and Applied and Interdisciplinary Informatics Group at the University of Kent under his supervisors, Nick Ryan and Jon Timmis.

Context Prediction based on Context Histories: Expected Benefits, Issues and Current State-of-the-Art

Rene Mayrhofer
Johannes Kepler University Linz
Altenbergerstr. 69
4040 Linz, Austria
+43 732 2468 8527
rene@soft.uni-linz.ac.at

ABSTRACT
This paper presents the topic of context prediction as one possibility to exploit context histories. It lists some expected benefits of context prediction for certain application areas and discusses the associated issues in terms of accuracy, fault tolerance, unobtrusive operation, user acceptance, problem complexity and privacy. After identifying the challenges in context prediction, a first approach is summarized briefly. This approach, when applied to recorded context histories, builds upon three steps of a previously introduced software architecture: feature extraction, classification and prediction. Open issues remain in the areas of prediction accuracy, dealing with limited resources, sharing of context information and user studies.

Keywords
Context prediction, context histories, time series prediction, machine learning

INTRODUCTION
Context histories, especially when recorded over a long term, offer a wide range of possibilities to enhance the services provided by some computer system. These possibilities include inferring of current and past user actions, selection of devices, etc. However, the prediction of future context based on the recorded past contexts is often conceived as the ultimate challenge in exploiting context histories. Context prediction, i.e. exploiting expected future context, can offer distinct advantages over the sole usage of past and current contexts: Obviously, it could be used to perform actions on behalf of the user, but this is problematic and will be discussed in more detail. However, it is also possible to exploit predicted context even without triggering actions in the physical world. On the one hand, comparing predicted contexts with recognized ones allows to detect irregularities and therefore assists in dealing with system failures. On the other hand, proactivity allows to provide user interaction that conforms better to the user's expectations. The efficiency of interpersonal communication builds upon a shared understanding of the past, current and last but not least future context within which interactions take place. Computer systems usually do not share such an understanding, and therefore at least a partial awareness of the relevant contexts is a prerequisite for a significant improvement of user interaction.

In addition to improving the human/computer interaction that is needed for most application areas, the introduction of proactivity opens new possibilities for automating application areas that are discussed in more detail in the next section. There are many examples from different areas that can benefit from an integration of proactivity: e.g. traffic and logistics (continuous planning and adaptation building upon estimated times of arrival, optimal utilization of road and parking place capacities, prevention of traffic jams), manufacturing (detection of and dealing with exceptions in just-in-time processes, planning for flexible manufacturing systems), individual traffic (prediction of arrival by the vehicle, warning before traffic jams, initializing or booting on board systems before they are used to prevent delays), medical care (alerting or initiating counter measures before critical situations can occur, digital dietary assistants that are aware of personal habits and predicted future events), communication (in-time establishment or change of connections, improved roaming, data synchronization and controlled shut down of sessions before connections are terminated), home automation (in-time establishment of custom room temperatures, reordering of groceries or fuel), etc. In combination with context awareness, proactivity opens numerous possibilities to enhance available informational services or construct new, currently unavailable ones. The next section presents a first taxonomy of applications that can benefit from context prediction.

Complementing the potentially large benefits of context prediction, there are serious issues with its technical, social and last but not least legal aspects. An overview of the currently perceived most important issues is given in a separate section. After listing the current issues, a first approach to context prediction based on recorded context histories is given and the remaining open issues are listed.

POTENTIAL BENEFITS OF CONTEXT PREDICTION

Although context prediction can be useful in most applications that currently utilize context awareness in general, a few application areas have been identified that benefit significantly from the introduction of proactivity. These application areas all focus on one central maxim: *to avoid potential problems caused by erroneous predictions*. As already mentioned in the introduction, an obvious benefit of context prediction is that it enables systems to perform actions on behalf of the user, like booking flight tickets when a potentially interesting conference will be held or ordering groceries when friends are invited to dinner. These two examples already indicate that automatically triggering actions based on context predictions is a delicate issue. What if the conference is indeed very interesting to the user and the system thus determines that she will attend it, but the budget does not allow for it? What if the invited group of friends decides spontaneously to go to a nearby restaurant instead? Predictions of future events will necessarily be imprecise, and in some cases they might even be impossible (cf. [5]). Therefore, we strongly suggest that systems that exploit context prediction should impose a design principle of not automatically triggering actions that can cause serious real world effects whenever a prediction is uncertain. Although the following areas of reconfiguration and accident prevention might influence real world objects, the effects of erroneous predictions tend to be limited. The following taxonomy of application areas that lend themselves to context prediction at the current state of research has first been presented in [4] and is summarized here:

Reconfiguration

System reconfiguration in general, not being restricted to context-based reconfiguration, is today one of the most time-consuming tasks associated with computer systems. We can further distinguish between *light-weight* and *heavy-weight reconfiguration*, where light-weight reconfiguration includes modification of the system configuration or general online, near real time adaptation to changed environments. Heavy-weight reconfiguration includes tasks that impose a noticeable delay during reconfiguration, leaving the system in question out of service until reconfiguration has finished. Boot-up of systems, installation or update of applications, maintenance and infrastructural changes, downloads, searches in large databases, etc. all consume, or even waste, significant amounts of valuable work-time. Any progress towards shortening these reconfiguration times yields a direct improvement for the involved people. We believe that such heavy-weight reconfiguration can be performed in advance by exploiting context histories to predict future context.

Accident prevention

An accident can be seen in the general case as an *undesirable system state*, and preventing such undesirable states has applications in many different areas. E.g. in telecommunication, an undesirable state is an overload in some network equipment or communication link. Load prediction is already used by larger telecommunication organizations to prevent system failures by proactively updating or bypassing highly loaded systems in time. In medical care, there is a vast multitude of undesirable or dangerous states and situations than can be monitored with bio sensors and should be predicted to prevent permanent damage. However, it is important to point out that with the approach presented in [4], such an undesirable situation must have already occurred in the past to be predicted. Lacking an application-specific model of desirable and undesirable situations – which can not be assumed when we focus on exploiting context histories – it is only possible to learn from past situations. Yet unknown contexts can not be predicted in a general, application-independent way. Thus, we strongly advise against using this approach for prediction and prevention of undesirable states in safety-critical systems.

Alerting

This is best known from the domain of PIM (personal information manager) type applications, including calendar, project management, scheduling, appointment and group coordination systems, but includes arbitrary applications that need to alert users in some form. These systems already provide a multitude of alerting capabilities, ranging from message boxes bound to being displayed on desktop computers or PDAs, signal lights, audible notification to sending emails, SMS or pager messages to user's mobile devices. However, events leading to such alerts are either triggered by certain actions (e.g. a colleague entering a virtual meeting room) or have been scheduled in advance, being entered in a calendar. When being able to predict future context, a device can autonomously issue alerts before some relevant contexts occur, without the need for manual scheduling.

Planning aid

Simply displaying predicted future context in a structured way and allowing to interactively browse it can provide a powerful aid for human-driven planning and scheduling. This puts people in the control loop, allowing to manually modify system behavior, but being assisted by predictions of future situations. Due to their informational nature, applications from this area will demand an estimation on the probability of the predictions being true, i.e. on their certainty, which might not be strictly necessary for other application areas.

In these application areas, context prediction can be exploited to provide better services to the user, but the effects of erroneous predictions should still be easy to undo or be even unnoticeable to the user because they can be reverted automatically. Until the certainty of predictions can be estimated satisfactorily to decide which predictions can be trusted and which can’t, this is an important feature. The following two sections present the aspects that need to be considered for context prediction, i.e. for all of the discussed application areas, as well as issues that appear when actually building such systems.

ASPECTS OF CONTEXT PREDICTION

There are many different aspects that need to be considered for context prediction, as it involves the recording of context histories, context recognition, time series prediction and acting on the real world based on those predictions. Initial experience with designing and implementing proactive systems shows that the following aspects are among the most important:

Time Series Aspects

The prediction engine should consider *sequential patterns*, *periodic patterns*, *long term trends* and possibly also *exceptions*. To enable the unsupervised recognition of periodic patterns, the length of the context history must include at least a few cycles of the longest period that should be detected. E.g., if seasonal effects should be predicted, a few years will need to be recorded continuously.

Training Aspects

Another important aspect is the kind of training, i.e. how the model is constructed. In machine learning, classification and prediction methods are usually distinguished as *supervised*, i.e. the target values are known for the training set, and *unsupervised*, i.e. only the input values are known.

This distinction should not be confused with the involvement of human experts in the training process, which is an orthogonal classification of approaches; such data mining approaches are sometimes also denoted as supervised approaches, although the involvement of users in the model construction is possible for supervised and unsupervised methods. For exploiting context histories, all options need to be considered, i.e. if the model is constructed automatically or via an interactive process involving human experts and if the approach is supervised or unsupervised. The associated issues are shortly discussed in the next section, but the appropriate selection of the training method is typically highly application dependent.

Context History Aspects

Additional aspects evolve rather around the recording of context histories than the usage of these histories for context prediction, but influence their exploitation and are consequently also discussed here. One of the most important aspects is the acquisition of ground truth, i.e. if "true" output values like user-specified context identifiers are recorded alongside the raw sensor data or not. Without such a ground truth, a quantitative assessment of the results is difficult, and often impossible (cf. [4]).

Another important aspect is the location of context histories, i.e. if they are stored in a centralized or a decentralized way. With decentralized storage, different costs of accessing other parts of the history arise and need to be considered in the usage of these histories. An example is the storage of short term histories locally at the system involved in the context prediction and long term histories on mass storage devices. Accessing long term history allows detecting periodical patterns of longer period, but involves higher cost.

The third aspect concerning the acquisition and recording of context histories, which is partially interrelated with the location, is the level of data that is recorded. There is a wide range of possibilities for context data on different levels, ranging from raw, unprocessed sensor data via data on the feature level to pre-classified context identifiers (cf. [4]). The higher the level of the recorded data, the less data typically needs to be stored, but the higher the effort for acquisition. Resource limited devices like nodes of a sensor network might even be incapable of the necessary pre-processing for recording data on any level other than raw sensor data. For context prediction, completely different approaches are necessary depending on the level of context data.

ISSUES

This section discusses issues of context prediction that emerged in most recent research on that topic and, for some of them, potential solutions or recommendations w.r.t. context histories. The following issues are likely to be present in nearly arbitrary uses of context histories:

- *Accuracy*: The accuracy of the recorded data is the factor with the highest influence w.r.t. the result quality. From the aims of a context history, the required accuracy and consequently the necessary sensor technology can be deduced.
- *Fault tolerance*: In real world experiments, missing values due to failing sensors and the inherent noise in sensor time series need to be dealt with. A more complex case are erroneous sensors that can not be detected directly as failing – which would allow to record missing values for the specific sensors – but that yield biased or completely erroneous values. These are more difficult to deal with than the "no-value" failures or the usual noise and often need sensor-level redundancy to compensate.
- *Unobtrusive operation*: Recording long-term histories, which are necessary for learning user behavior from scratch, i.e. without expert knowledge, or recording data from multiple users requires an unobtrusive operation that does not interfere with the normal activities of the test subjects. This is necessary both for the recording and for the usage of context histories in practical applications.
- *User acceptance*: In every exploitation of context histories and often even in their recording, a felt loss of control of involved users is a serious problem that currently needs to be addressed in an application specific way, which might include non-technical means like user education or organizational changes.
- *Privacy*: Closely related to the previous issue is the area of privacy – including legal aspects that are still to be clarified. Privacy issues might also lead to problems with user acceptance, but typically only few users are aware of the implications of recording extensive context histories.

Therefore, privacy issues must be tackled by the designers of experiments and systems that record context histories.

More specific to context prediction are the following issues:

- *Supervised vs. unsupervised*: Only when ground truth is available, supervised learning methods can be used. With unsupervised methods, evaluation of results is more difficult. For context prediction, ground truth for training purposes can be extracted from any recorded context trajectory, i.e. context time series, by splitting the trajectory into a training and a test set. For evaluating the prediction results, the test set can be used. When context prediction is used in online systems, ground truth can not be known immediately, but becomes available when the predicted time has passed.
- *Automatic vs. manually assisted*: For small data sets, human experts can construct the respective prediction model, possibly assisted by data mining techniques or suggestions by the system. This expert-driven approach is only feasible for few experiments, but usually not for the independent prediction of the contexts of many users. In this case, the prediction model needs to be constructed automatically, based solely on the available context history and potentially some domain specific knowledge that has been embedded into the learning process.
- *Problem complexity*: A serious issue is the general complexity of time series prediction problems w.r.t. the size of the recorded data sets and run-time complexity for constructing the models and subsequently determining predictions based on the models. Most of the more powerful prediction techniques bring forth considerable demands for processing and storage capabilities and might thus be unsuitable for embedded or mobile systems.
- *Uncertainty*: Using uncertain predictions to act on the real world is generally problematic, as discussed in more detail above; if there is any doubt about some prediction of future context – and in almost any cases doubt is expedient in time series prediction – then it is recommended to "play safe" and not to depend on the predictions for critical actions. It is generally advisable to leave the user in the control loop (cf. [1]).
- *Online processing*: If context prediction – or in fact any exploitation of context histories – should be embedded into computer systems in the spirit of pervasive computing, it needs to happen online, without a distinction into training and usage phases. A device must be continuously available and must be adaptive to changing environments. This makes it impossible to use some learning methods that depend on batch training.
- *Heterogeneity*: Values gathered from typically available sensors are highly heterogeneous and thus many algorithms for statistical analysis and classification are not directly applicable.

CURRENT APPROACHES

The current approach to context prediction suggested in [4] is the prediction of abstract contexts – in contrast to the autonomous prediction of individual aspects like the geographical position of the user. It is based on a multi-step software architecture that separates context recognition, i.e. the classification of raw sensor data to higher-level context identifiers, from context prediction, which is based on the trajectories of context identifiers. This approach is inherently decentralized, because each device is supposed to recognize and predict context independently. By distributing the acquisition and exploitation of context histories, privacy issues are mostly avoided, as long as the personal device that records and predicts context is trusted. Privacy and the issue of limited resources are also supported by the use of online methods as far as possible. This way, only a sliding window of the context history needs to be available instead of the complete time series data. [4] gives an overview of prediction methods suitable for context prediction. Particular attention is turned to implicit user interaction to prevent disruptions of users during their normal tasks and to continuous adaptation of the developed systems to changing conditions. Another considered aspect is the economical use of resources to allow the integration of context prediction into embedded systems. The developed architecture has been implemented as a flexible software framework and evaluated with recorded real-world data from everyday situations.

Other approaches that have not yet been studied in detail by the author are to predict sensor or feature level data instead of context identifiers, and to record and use the complete context history. The former has the advantage that domain-specific knowledge about the sensors can be exploited for prediction, e.g. geographical maps for location prediction, but the disadvantage that correlations between different sensors are not considered in the prediction model. This is also an issue for many other uses of context histories, since the interrelations between different sensor time series are often not apparent at the lower levels of raw sensor data but need to be recognized by applications using those histories. The latter approach allows the usage of more powerful prediction methods, but imposes significantly higher demands on the storage capabilities of involved devices, which is again not limited to context prediction but is true for arbitrary uses of context histories.

CONTRIBUTION

The present position paper discusses potential benefits, aspects and current issues of context prediction and briefly summarizes a first approach as presented in more detail in [4]. This approach considers – and partially addresses – the issues of fault tolerance, unobtrusive operation, privacy, unsupervised context recognition, automatic construction of the prediction model, online processing, and heterogeneity. The issue of uncertainty is shifted to applications implemented on top of the developed architecture, but dealing with it at application level is

assisted by providing measures of certainty of recognized and predicted contexts. Both major parts of this architecture, i.e. the recognition and the prediction parts, can be used independently and thus allow to record and to exploit context histories in terms of context recognition. Particular contributions of the architecture are to enable a continuous, unsupervised learning of user behavior with life-long adaptation to changing environments and the use of nominal and ordinal sensor values in addition to numerical ones, effectively solving the issue of heterogeneity.

Context prediction is only one possible use of context histories, even if it might be the most challenging one. Nonetheless, there are many other uses that can provide additional benefit to the user of a system and that have already been analyzed in more depth by current research. Many of the issues discussed in this paper are also valid for other uses of context histories, and might thus be of help to research on those applications as well.

OPEN CHALLENGES

Most of the discussed issues are addressed by the presented approach to context prediction, but few are solved completely. Open challenges remain especially in:

- *Improving the accuracy of predicted contexts*: The time series prediction methods considered so far address sequential pattern prediction, but lack a detection of arbitrary periodical patterns and long term trends. Current developments like the algorithm presented in [2] to detect periodicities should be examined w.r.t. context prediction.
- *Coping with limited resources*: By applying online methods to context recognition and prediction, required storage and processing resources are generally low. However, eviction policies that are necessary to deal with strictly limited memory or real-time issues have not yet been considered.
- *Sharing context histories between devices*: As also mentioned in [3], sharing of context information can improve the accuracy of context recognition, and subsequently context prediction, by enhancing the view of the environment of each device with information that is not available locally. This is not necessarily limited to sharing only current context information, but could be extended to sharing complete context histories and predicted contexts.
- *Unobtrusive user interfaces for labeling context identifiers*: In our current work, a mapping of automatically recognized context classes, i.e. automatically constructed higher-level context identifiers, to descriptive context labels assigned by the user is assumed to be handled by the application. It is an open issue for HCI to design appropriate user interfaces for assisting this interactive process in an unobtrusive manner.
- *User acceptance*: No empirical user studies w.r.t. user acceptance of continuous context prediction have been conducted so far, but will be necessary before context prediction systems can be put into service for end users.
- *Uncertainty*: Dealing with uncertain sensor information on the one hand and with uncertain predictions on the other hand is currently not addressed satisfactorily. It is still an open issue for most time series prediction methods to compute measures of certainty alongside the actual predicted values.

Context prediction is a young topic, still at the outset of methodical research. When applied in a way that still leaves users in the loop of control, it can be a powerful tool to support users in their daily lives and to foster a broad availability of computing services to a larger public. However, the social implications of pervasive computing, and more specifically of exploiting context histories, must not be neglected; not only technological, but more importantly non-technological issues like a felt loss of control will rather sooner than later become urgent concerns.

ACKNOWLEDGMENTS

The author thanks Alois Ferscha for his initial ideas and several discussions about context prediction which helped to shape and clarify the concepts and the resulting software architecture. These concepts, the resulting software architecture and its implementation as a software framework have been developed in close cooperation with Harald Radi.

REFERENCES

1. Barkhuus, L., and Dey, A. Is Context-Aware Computing Taking Control away from the User? Three Levels of Interactivity Examined. In *Proceedings of the 5th International Conference on Ubiquitous Computing (UbiComp'03)*, 149—156, October 2003
2. Elfeky, M.G., Aref, W.G., and Elmagarmid, A.K. Using Convolution to Mine Obscure Periodic Patterns in One Pass. In *Proceedings of the 9th International Conference on Extending Database Technology (EDBT)*, 605—620, March 2004
3. Mäntyjärvi, J., Himberg, J., and Huuskonen, P. Collaborative Context Recognition for Handheld Devices, In *Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications (PerCom'03)*, 161—168, March 2003
4. Mayrhofer, R. An Architecture for Context Prediction. *Trauner Verlag*, Schriften der Johannes-Kepler-Universität Linz, volume C45, April 2005. (first appeared as *PhD Thesis*, Johannes Kepler University Linz, October 2004)
5. Penrose, R. The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics. *Oxford University Press*, 1989

Author Biographies

Rene Mayrhofer received his Dipl.-Ing. degree in Computer Science in 2002 and his PhD in Technical Sciences in 2004, both from the Johannes Kepler Universität Linz, Austria. He is currently working at the institute for Pervasive Computing as an assistant. His research interests include context awareness, embedded systems, peer-to-peer networks, artificial intelligence with specialization on spiking neural networks and security. He has served on the committees of PERVASIVE 2004 and PERVASIVE 2005 and is a contributing developer of the Debian GNU/Linux project.

A Stochastic Approach for Creating Context-aware Services based on Context Histories in Smart Home

Hua Si, Yoshihiro Kawahara, Hiroyuki Morikawa, Tomonori Aoyama
Aoyama Morikawa Lab, Hongo Campus Bldg #3.
Department of Information and Communication Engineering, Faculty of Engineering,
The University of Tokyo 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656 JAPAN
+81 3 5841 6710
{sihua, kawahara, mori, aoyama}@mlab.t.u-tokyo.ac.jp

ABSTRACT
In this paper, we present a context-aware service platform called Synapse and its simple smart home test bed. By exploiting the recorded histories of contexts and services, Synapse can learn different users' habits. Then Synapse can predict the most relevant services that users will use in the current situation based on their habits, and provide services in Active Mode and Passive Mode. Considering the challenges faced, we apply a stochastic approach – Bayesian Networks [17] to build the model of Synapse, and implement a flexible, end-user manageable system, which can absorb various uncertainties from multi-dimensional sensor data and provide personalized services.

Keywords
Context-aware service, HMM (Hidden Markov model), context histories, smart home

INTRODUCTION
Context-awareness is now regarded as a key ingredient for pervasive computing, and several toolkits such as Context Toolkit [21] and Location Stack [4] have been proposed to incorporate users' contexts into network applications. An attractive category of context-aware applications is the service automation in the indoor environment such as home and office [15], which aims at providing users with dynamic services that adapt to changing environment on the basis of users' habits. Obviously, the best ground for learning users' habits exists in the recorded histories of the users' interactions in context (context histories for short). The usefulness of location history has been explored to report users' mobility patterns in an office [13]. However, in the real world applications, users' contexts contain diversity, from users' activities to environmental status; and users' habits vary widely, from the usage of services to the mobility patterns. Therefore, a dynamic mechanism is necessary, which can provide various services by taking the users' contexts into account. We are developing a context-aware service platform – Synapse, which can learn different users' habits by exploiting the recorded histories of contexts and services, then predict and provide the most relevant services that users will use in the current situation based on their habits. We are implementing a smart home test bed of Synapse, and three simple scenarios are used to examine the practicability of our methods.

The following "daily scenarios" are assumed to be learned from users' context histories, and are used for evaluation:

- **The "Light" Scenario**: if it is too dark in the room, Synapse will automatically turn on the light.
- **The "TV" Scenario**: Synapse will recommend TV programs appropriate for people in the living room. (If only kids are watching TV, cartoon videos will be recommended. When parents and kids are watching TV together, Discovery or some other channels will be recommended.)
- **The "Music" Scenario**: Synapse will automatically turn down the volume of the music player when someone is using the phone, and turn up the volume after using it.

We faced several challenges when we designed our system. First, considering the flexibility of system and the ease of management for end-users, we should apply a dynamic mechanism rather than binding the contexts and services in a specification language such as ECA [12]. Second, since users' habits may slowly change as time advances, our algorithms should have the ability of updating to reflect it. Third, corresponding to the diverse contexts (such as "the user is sitting", "the brightness in a room") and various services (such as "turn on light", "select TV channel 3"), our model should have the capability to deal with multi-dimensional inputs and outputs. Fourth, personalized services are desired by different users. Finally, the system should work with imperfect and noisy sensor data.

With these challenges in mind, we apply a stochastic approach for Synapse, which is based on one of Bayesian Networks [17] – HMM (Hidden Markov Model) [18]. The model of Synapse consists of continuous cycles. Each cycle is composed of two phases: Learning Phase and Executing Phase. In the Learning Phase, Synapse learns the relationship between contexts (we call them "sensor events" in Synapse) and services by exploiting the recorded histories of them. Then in the Executing Phase, based on the learned relationship and the current sensor events,

Synapse predicts the most possible services to be used and provides them to users. Since users would like to enjoy autonomous services in a moderate degree without losing control of them [2], Synapse provides services in two modes: Active Mode will start a service automatically based on sensor events, while Passive Mode recommends the top 5 relevant services in a list and let users select. The results of the Learning Phase are used as prior knowledge for the next cycle. To easily achieve personalization, user ID is treated as a sensor event.

The related works of time-series prediction and smart home projects will be introduced in section 2. The architecture of Synapse will be explained in section 3. The preliminary evaluation of Synapse will be discussed in section 4. The conclusion and future work will be given in section 5.

RELATED WORKS

For time-series prediction of continuous data, linear models (such as ARIMA, ARMAX [5]) or non-linear models (such as neural networks or decision trees [14]) are usually used. For discrete data, n-gram models [8] or variable-length Markov models [20] are common choices. Compared to these methods, Dynamic Bayesian Networks (DBN) [17] have some advantages appropriate for the challenges we face: First, it is easier for DBN to deal with multi-dimensional inputs and outputs. Second, prior knowledge is easy to be incorporated, so the prediction of the future is based on all the past history. Third, DBN is more flexible than simple supervised classifiers. Finally, DBN has been successfully used in many areas [6, 11, 19] for time-series prediction.

Therefore, we choose HMM [18], one of Bayesian Networks, to build the core model of Synapse. This core model is a general context-aware platform, which can be used not only in smart home environment, but also in a broad range of context-aware applications that need to correlate the contexts and services, since it provides standard interfaces for contexts and services.

Several smart home projects are in progress. The Georgia Tech Aware Home [1] and MIT House_n [7] use an array of sensors to determine users' locations and activities within an actual house. The Neural Network House [16] balances the goals of anticipating user needs and energy conservation through a neural network. The MavHome [3] uses an intelligent and versatile home agent to perceive the state of the home through sensors and act on the environment through effectors. The industrial examples are also available, such as the Microsoft Easy Living project, the Cisco Internet Home, and the Verizon Connected Family project. Although, similar with these projects, our smart home test bed of Synapse extracts contexts from raw sensor data and adopts services from smart devices, our original core model guarantees the uniqueness of Synapse.

ARCHITECTURE OF SYNAPSE

The smart home test bed of Synapse consists of four parts: 1) the sensor event collection part that captures real world information, 2) the service control part that provides services, 3) the Synapse Core, and 4) the user interface. Architecture of Synapse is shown in Figure 1.

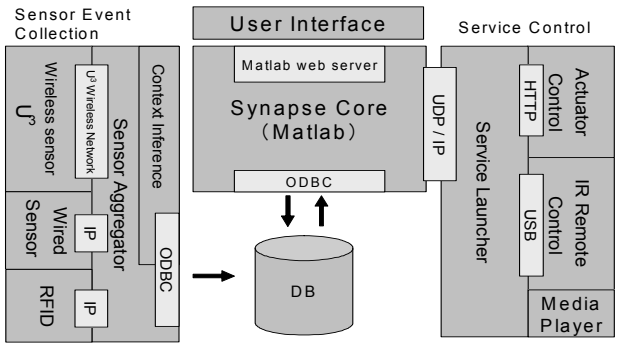


Fig. 1: Architecture of Synapse.

Sensor Event Collection Part

The sensor event collection part captures real world information from various sensors, converts raw data into useful contexts (we call them "sensor events" in Synapse), and records these sensor events in database. Sensor Aggregator fuses the raw sensor data and reduces the noise. For instance, the average temperature in a room is fused from different temperature sensors. Context Inference extracts complex events such as "the user is sleeping" from simple events. On this test bed, 4 kinds of sensors are used to produce 11 events: RFID is used to identify users, U^3 wireless sensor nodes [9] are used to capture the temperature, brightness and human motion, a contact detector detects whether the phone is in use, and an e-calendar detects a day of the week.

All the sensor events are recorded as a time series $\{E_1, E_2 \ldots\}$ in database. Each sensor event is recorded as E (EN, EV, ET), which respectively represents the event ID, the event value and the time at which this event is recorded. We predefine a set of events $\{e_1, e_2 \ldots e_N\}$ (such as e_1 means "temperature", e_2 means "brightness"), and $EN \in \{e_1, e_2 \ldots e_N\}$. Many context inference schemes can be used to recognize events and their values from raw sensor data [10]. However, since event values are generated from different types of sensors (e.g. the temperature is 25°C, and the humidity is 60%), and it is difficult for a general core to process all types of values, we use fuzzy sets [22] approaches to unify all the event values between 0 and 1 as in [10], which means $EV \in [0, 1]$. Basically, an event will be recorded when the value changes. However, in many scenarios, it is not necessary to record events as frequently as they change, so we can add some requirements to event recording. Events will not be recorded, until they satisfy these requirements. (e.g. one requirement is "e_1 is over 0.7", so e_1 will not be recorded until it is over 0.7.)

Service Control Part

The service control part controls various devices to supply services. Service Launcher operates as a proxy between Synapse Core and the devices. It can receive a service ID from Synapse Core through UDP/IP networks, and controls the device corresponding to this service ID. It can also send the ID of a selected service to Synapse Core for service recording. As a result, it is easy for Synapse to add new services, since Synapse Core can manage them with only IDs, and ignore the various operations of different devices. On this test bed, 4 devices are used to provide 23 services: a light and a fan provide on/off services, a TV provides on/off, 12 channels and 2 videos, and a music player provides on/off and music mute/loud services.

All the services are recorded as a time series $\{S_1, S_2 \ldots\}$ in database. Each service is recorded as S (SN, ST), which respectively means the service ID, and the time at which this service is recorded. We predefine a set of services $\{s_1, s_2 \ldots s_M\}$ (such as s_1 means "turn on light", s_2 means "mute music"), and SN $\in$ $\{s_1, s_2 \ldots s_M\}$.

Synapse Core

We apply HMM to model the relationship between the sensor events and the services. Figure 2 shows one cycle of Synapse model. There are two basic components in HMM: the hidden state X_t and the observation of state Y_t. In Synapse, each hidden state X_t corresponds to a service S_t (*not lowercase s*), to indicate the situation in which this service is used, and the observation Y_t is a vector of event values $(y_1, y_2 \ldots y_N)$, which are the current values of $\{e_1, e_2 \ldots e_N\}$. There are three parameters in HMM: the prior probability $\pi(i)=P(X_1=i)$ which represents the initial state, the transition matrix $A(i,j)=P(X_t=j|X_{t-1}=i)$ which represents the probability of transfer from $X_{t-1}=i$ to $X_t=j$, and the observation model $P(Y_t|X_t)$ which represents the relation of X_t and Y_t [16]. The learned results in one cycle are used as the initial estimations of the next cycle.

In the Learning Phase ($1 \le t \le T$), Synapse uses the history records of sensor events and services that happened during t=1, 2…T to compute $A(i,j)=P(X_t=j|X_{t-1}=i)$ and $P(Y_t|X_t)$. We assume that sensor events, which happened in a certain interval before a service, indicate the situation in which this service is used. For instance, in Figure 2, E_2 and E_3 indicate the situation in which S_2 is used. Since X_t cannot be observed directly, we firstly use the forwards-backwards algorithm [17] to infer $X_{1:T}$ from the observation $Y_{1:T}$. In the forwards pass, we recursively compute the filtered estimate $\alpha_t=P(X_t=i|Y_{1:t})$, and in the backwards pass, we recursively compute $\beta_t=P(Y_{t+1:T}|X_t=i)$; then combine them to produce the smoothed estimate $\gamma_t(i)=P(X_t=i|Y_{1:T})$ and the smoothed two-slice estimate $\xi_{t-1,t}|T(i,j)=P(X_{t-1}=i,X_t=j|Y_{1:T})$. After that, we use EM (expectation maximization) algorithm [17] to learn $A(i,j)=P(X_t=j|X_{t-1}=i)$ and $P(Y_t|X_t)$ from $\gamma_t(i)$ and $\xi_{t-1,t}|T(i,j)$.

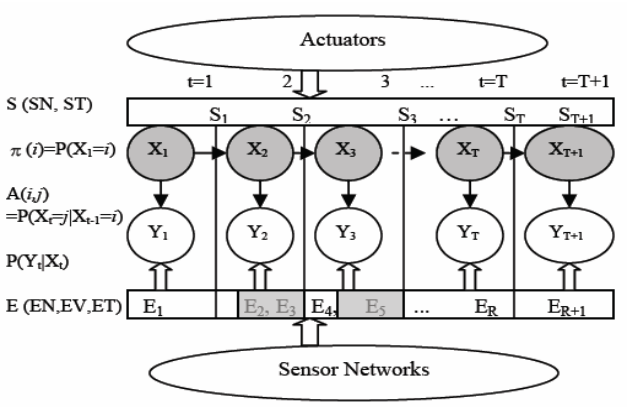


Fig. 2: One Cycle of Synapse Model. The grey rectangles indicate a certain interval before S_t.

In the Executing Phase (t>T), Synapse uses the learned transition matrix $A(i,j)=P(X_t=j|X_{t-1}=i)$, observation model $P(Y_t|X_t)$ and the current observation Y_t to compute the occurrence probability of each service. A two-step filtering algorithm is applied: in update step, the probabilities of current state can be gained as we compute $P(X_t|Y_t)$; in predict step, the probabilities of next state can be predicted as we compute $P(X_{t+1}|Y_t)$. As a result, the occurrence probability of each service can be computed as the occurrence probability of each state corresponding to these services. After that, we can sort the services in a descending order of probability. If a probability is higher than a user-defined threshold, the corresponding service will automatically start in Active Mode. The top 5 services will be recommended as a list to the user interface in Passive Mode. Passive Mode is mainly used in Synapse. All these algorithms are implemented on Matlab.

User Interface

Synapse provides a user interface in XML form on Matlab Web Server. Users can browse this web through PC, PDA, or cellular phone, and start a service by selecting the service ID. The recommended service list on this web can automatically update after a fixed interval, or be manually updated by users.

PRELIMINARY EVALUATION OF SYNAPSE

In order to examine the practicability of our methods, we implemented three simple scenarios on the smart home test bed, and preliminarily evaluated Synapse on three aspects: 1) feasibility of Synapse, which means whether Synapse can successfully provide services based on the learned habits and the current sensor events, 2) time complexity of algorithms, which examines whether it is practically quick enough to gain the results, 3) correctness of the recommendation, which examines whether the results of prediction are practically accurate enough.

Feasibility of Synapse

Using the sensors and devices mentioned in section 3, we collected 200 training samples: 40 of which are "Light"

scenario using "Light_On" service, 80 of which are "TV" scenario using "Video" and "TV_1ch" services (40 respectively), and 80 of which are "Music" scenario using "M_Mute" and "M_Loud" services (40 respectively). Each sample is a combination of one service and a group of sensor events. For instance, in "Light" scenario, when a user was in the room and it was too dark, he selected "Light_On" service, so the user's ID, the brightness and the "Light_On" service were recorded as one training sample. We used such training samples to learn users' habits in three scenarios.

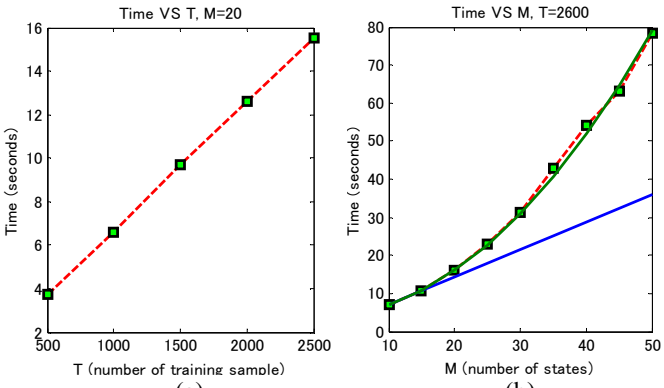
After learning, we changed the status of users and environment, and Synapse successfully provided dynamic services adapting to the changed situation. For instance, in "Music" scenario: when we was using the phone, Synapse provided "M_Mute" to turn down the volume of the music player; when we finished using the phone, Synapse provided "M_Loud" to turn up the volume. These were collected as test samples, which were used to examine the correctness of recommendation.

Time Complexity of Algorithms

We simulated the time complexity of algorithms on Matlab. The number of hidden state – M and the number of training sample – T are important to estimate the complexity.

In the Learning Phase, if there are M hidden states, it will take $O(M^2)$ operations at every time slice, since we must do several matrix-vector multiplies per time slice. And as we must repeat this procedure during t=1, 2…T, it will totally take $O(M^2T)$ time. In the Executing Phase, algorithms do approximately the same work as learning algorithms do at one time slice. Therefore, the time complexity is $O(M^2)$. Figure 3 depicts the time complexity of learning algorithms.

Figure 3 shows that for 50 states and 2600 training samples, it takes approximately 80 seconds to learn the parameters, which reveals that our methods are practically quick enough for real life.



(a) (b)

Fig. 3: (a) shows the relation of Time and T. (b) shows the relation of Time and M. The green curve is quadratic, and the blue one is linear.

Correctness of Recommendation

We verified the correctness of recommendation with 150 test samples: 30 of which are "Light" scenario test samples, 60 of which are "TV" scenario test samples (30 for Video, 30 for TV_1ch), and 60 of which are "Music" scenario test samples (30 for M_Mute, 30 for M_Loud). We only tested correctness of the first recommendation because of the definitude of result. (The correctness of top 5 recommendations will be tested by real inhabitants in the future.) The correctness of recommendation is shown on Table 1, which reveals that our methods are practically accurate enough for real life.

Table1: Correctness of Recommendation

Services	Light_On	Video	TV_1ch	M_Mute	M_Loud
Correct	96.7%	93.3%	90.0%	93.3%	90.0%

CONCLUSION AND FUTURE WORK

In this paper, we presented a context-aware service platform – Synapse and its smart home test bed. By exploiting the recorded histories of contexts and services, Synapse can learn the users' habits. After that, Synapse can predict the most relevant services that users will use in current situation based on their habits, and provide services in Active Mode and Passive Mode. We described our algorithms and the implementation of smart home test bed in detail. The preliminary evaluation with real world data revealed that Synapse was practicable and should be built at home.

Now we are extending the sensor and service parts and implementing an entire Synapse system in a house. The experiment with real inhabitants will be conducted in the future.

ACKNOWLEDGMENTS

This work is supported by Ministry of Public Management, Home Affairs, Posts and Telecommunications.

REFERENCES

1. Aware Home. http://www.cc.gatech.edu/fce/ahri/.
2. Barkhuus, L. Is Context-Aware Computing Taking Control Away from the User? *Proceedings of Ubicomp 2003*, LNCS 2864.
3. Das, S. The Role of Prediction Algorithms in the MavHome Smart Home Architecture. *IEEE Wireless Communications*, vol. 9, no. 6, pp. 77-84, Dec. 2002.
4. Graumann, D., Lara, W. Real-world implementation of the location stack: The universal location framework. *Proceedings of WMCSA 2003*, 122–128.
5. Hamilton, J. *Time Series Analysis*. Wiley, 1994.
6. Horvitz, E. The lumiere project: Bayesian user modeling for inferring the goals and needs of software users. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*. 1998.

7. House_n. http://architecture.mit.edu/house_n/.
8. Jelinek, F. *Statistical methods for speech recognition*. MIT Press, 1997.
9. KAWAHARA, Y. Design and Implementation of a Sensor Network Node for Ubiquitous Computing Environment. *Proceedings of IEEE Semiannual Vehicular Technology Conference*, 2003.
10. Korpipä, P. Bayesian approach to sensor-based context awareness. *Personal and Ubiquitous Computing*, Vol. 7 Issue 2, July 2003.
11. Korvemaker, B. Predicting UNIX Command Lines: Adjusting to User Patterns. *National Conference on Artificial Intelligence 2000*, AAAI press, 230-235.
12. López, D., Katsiri, E. An ECA Rule-Matching Service for Simpler Development of Reactive Applications. *IEEE Distributed Systems* 2001, Vol. 2.
13. Mantoro, T. Location history in a low-cost context awareness environment *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003,* Vol. 21.
14. Meek, C. Autoregressive tree models for time-series analysis. *Proceedings of the Second International SIAM Conference on Data Mining 2002*, 229–244.
15. Meyer, Sven. A survey of research on context-aware homes. *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003*, Vol. 21.
16. Mozer, M. An intelligent environment must be adaptive. *IEEE Intelligent Systems*, vol. 14, no. 2, pp. 11-13, Mar. /Apr. 1999.
17. Murphy, K. Dynamic Bayesian Networks: Representation, Inference and Learning. PhD Dissertation, UC Berkeley, 2002.
18. Rabiner, L. R. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE* 1989, 77(2):257–286.
19. Rao, S., Cook, D. J. Predicting Inhabitant Actions Using Action and Task Models with Application to Smart Homes, *International Journal of Artificial Intelligence Tools*, 13(1), 81-100, 2004.
20. Ron, D., Singer, Y. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning* 1996, 25.
21. Salber, D., Dey, A.K. The context toolkit: Aiding the development of context-enabled applications. Technical report (2000), Georgia Institute of Technology.
22. Zadeh, L.,"Fuzzy sets", Information and Control 8:338-353, 1965.

BIOGRAPHIES

Hua Si received the B.E. in Electronic Techniques and Information Systems from Tsinghua University, Beijing, China, in 2002. He is currently a master student of the Graduate School of Information Science and Technology at the University of Tokyo with an emphasis on ubiquitous middleware applications and machine learning. He is a student member of IEEE and IEICE.

Yoshihiro Kawahara received the B.E., M.E., and Dr. Eng. degrees in Information Science and Technology from the University of Tokyo, Tokyo, Japan, in 2000, 2002, and 2005, respectively. He is currently a Research Associate of the Graduate School of Information Science and Technology at the University of Tokyo. His research interests are in the areas of context-aware computing, computer networks, and wearable computing. He is a member of IEEE, IEICE, and IPSJ.

Hiroyuki Morikawa received the B.E., M.E., and Dr. Eng. degrees in electrical engineering from the University of Tokyo, Tokyo, Japan, in 1987, 1989, and 1992, respectively. He is currently an Associate Professor of the Department of Frontier Informatics at the University of Tokyo. From 1997 to 1998, he stayed in Columbia University as a visiting research associate. His research interests are in the areas of computer networks, ubiquitous networks, mobile computing, and wireless networks. He serves as Editor of Transactions of Institute of Electronics, Information and Communication Engineers (IEICE) and on the technical program committees of IEEE/ACM conferences and workshops. He is a member of IEEE, ACM, ISOC, IPSJ, and ITE.

Tomonori Aoyama received the B.E., M.E. and Dr. Eng. from the University of Tokyo, Tokyo, Japan, in 1967, 1969 and 1991, respectively. Since he joined NTT Public Corporation in 1969, he has been engaged in research and development on communication networks and systems in the Electrical Communication Laboratories. From 1973 to 1974, he stayed in MIT as a visiting scientist to study digital signal processing technology. In 1994, he was appointed to Director of NTT Opto-Electronics Laboratories, and in 1995 he became Director of NTT Optical Network Systems Laboratories. In 1997, he left NTT, and joined the University of Tokyo. He is currently Professor in the Department of Information and Communication Engineering, Graduate School of Information Science and Technology, the University of Tokyo. His research activities cover the next generation networking technologies from layer 1 (e.g. photonic networks) to higher layers including middleware for network collaboration, P2P routing, mobile networking, and ubiquitous networking. Dr. Aoyama is involved in several governmental projects such as Japan Gigabit Network (JGN) and the Ubiquitous Networking Forum, and in some non-profit organizations and consortiums such as the

Photonic Internet Forum (PIF) and the Digital Cinema Consortium (DCC) in which he is serving as Chairman. Dr. Aoyama is IEEE Fellow and was a Members-at-Large of the IEEE ComSoc Board of Governors. He served as President of IEICE Communication Society. He also served Chair of IEEE ComSoc Japan Chapter. He is a member of IPSJ and IEEE.

Building a Personal Memory for Situated User Support

Michael Schneider, Mathias Bauer, Alexander Kröner

German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbruecken, Germany
{michael.schneider, mathias.bauer, alexander.kroener}@dfki.de

ABSTRACT
Keeping a history of the user's interaction with the environment is of use for many reasons. However, collecting, structuring, accessing, and reviewing such potentially large amounts of information is not trivial. In this paper we present our ideas for a memory model for pervasive computing applications addressing these questions. The proposed architecture allows applications to deliver ad-hoc support taking into account the user's history and general attitudes as well as providing a personal diary to review events and retrieve memories. We also include a brief discussion of a novel user interface, which allows the user to bind services to general contexts based on her previous experiences.

Keywords
Context histories, user modeling, adaptive user support

INTRODUCTION
The diffusion of sensor technology from dedicated devices into our everyday environment offers a potentially omnipresent, rich source of information that might be used by pervasive computing applications in multiple ways. An example of such an application is an artificial memory extending the user's perception. With such a memory, on the one hand context dependant support can be provided to the user by considering experiences previously made in similar situations. On the other hand, such an artificial memory could complement the user's natural memory and could be used to retrieve forgotten or unnoticed information at a later point.

On the way to this ambitious goal three main research questions arise:

1. How is useful information identified and acquired?
2. How is stored information structured/organized?
3. How is memory content retrieved and reviewed?

All the work described in this paper is conducted within the project SPECTER. Goal of the project is to build a personal ubiquitous assistant, which keeps an artificial memory of the user's experiences in order to deliver ad-hoc and subsequent context dependant support. As such, SPECTER has to deal with all of the above questions. However, in this paper, we focus on the second question and present some concepts related to the third question. For reasons of completeness we will include a rather short and practical discussion of the first question.

Since the project SPECTER is still running, not all of the ideas presented in this paper have been fully implemented yet. Therefore, we will give implementation details where possible and discuss our theoretical ideas otherwise.

The rest of this paper is structured as follows. After a description of the demo scenario used for the examples in this paper, we present a practical approach to the information acquisition problem. We continue with the description of the memory model used in SPECTER to store and organize the user's experiences, which we apply for building memories from perceptions and for user support. In the sequel, we describe how the user may apply the artificial memory in order to configure the user support. After a description of related work we conclude with a summary of our results and an outlook of future work.

SCENARIO
Our demo scenario is about a user preparing her shopping trip at home using the World Wide Web, moving to a real world shop, and executing actions in the shop like inspecting and comparing multiple products. Back at home, the user reviews her shopping trip with assistance of the SPECTER system and provides additional information where necessary. This information may be provided by the user on her own free will, or may be requested by the system (e.g., to gather feedback about a service which was suggested by the system, but was rejected by the user).

Multiple sensors are used in this scenario: At home, a special proxy software [13] observes the user browsing the WWW and especially e-commerce sites like Amazon. In the real world shop, the user's actions are recognized by RFID-equipped shelves and products. Additional context information is acquired through web services. Currently we are using weather information and detailed product information (provided by [1]). The only sensor owned by the SPECTER system is the location sensor, which is based on a hybrid system using GPS outside and IR transmitter and active RFID tags inside buildings (cf. [4]).

The system is involved in the described scenario in diverse ways. We address in this article two of them: The recording and analysis of the user's experiences during her shopping trip, and the application of this information for triggering services as part of the user support. Such services may range from management of advertisements over assistance in a product comparison to suggestions for a coffee break. The foundation of this mechanism is a binding between

services and situations, which is defined by the user in a collaborative process with the system. This issue leads to another topic addressed in a later section—the question of how a user may specify situations by using the system's memories without being overwhelmed by the sheer amount of recorded information.

INFORMATION ACQUISITION

For the acquisition of relevant sensor information we take a rather practical approach. We expect sensors to publish their information as some kind of location-based service within the area they can "observe". Following this idea, an instrumented shelf would for instance publish its sensor information to devices located in front of that shelf. In principle, the granularity of this approach depends only on the resolution and accuracy of the positioning sensor. As the focus of our project is on the memory structures, we used a slightly simpler approach in our demo implementation based on a hard coded sensor registry published on the local (wireless) subnetwork.

Out of the potentially manifold information sources present in an environment, we consider by default only those providing information with a well defined and machine understandable semantic. This especially includes information represented as instances of concepts defined in some ontology (discussed in more detail in the next section) and in general excludes audio and video data. This is due to the circumstance that we later want to apply automated memory processes on the incoming information, which is hard to do without defined semantics. However, the user could choose at any time to manually add audio or video information to her records.

The sensors in the environment support two modes for information access: Pull and Push. When entering a new context, the SPECTER system first acquires the current status from the newly discovered sensors by use of the pull mechanism. Subsequently, the push mechanism is applied to notify the SPECTER system about observed changes and events in the environment.

MEMORY MODEL

In the following we will describe our application framework with a focus on the memory model responsible for the recording of and the reasoning about the user's experiences.

An overview of our framework and the employed memory model is given in Figure 1. In general, data provided by an instrumented environment is at first collected in a short-term memory to form a snapshot of the user's current context. Support may be delivered in this stage by firing context-aware service triggers previously defined by the user. As the users moves on, outdated information stored in the short-term memory is transferred into the long-term memory. The content of the long-term memory can later be reviewed and evaluated by the user in a process called introspection. The long-term memory provides support based on a user model learned from the evaluated long-term memory content. As such the model is supposed to reflect the user's general attitudes and preferences.

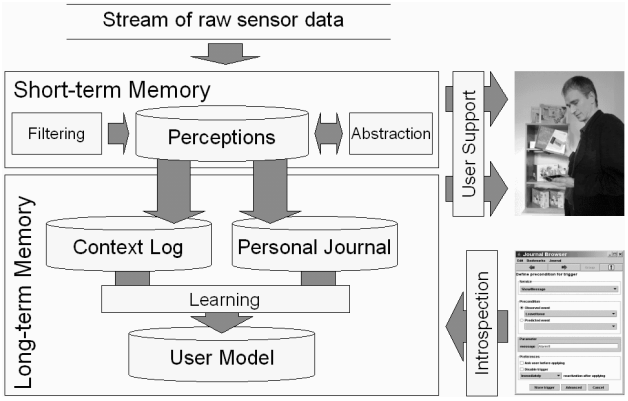


Figure 1: SPECTER's memory model: from low-level perceptions to introspection and user support

Our design was guided by psychologists' research (cf. [10]) on the structure of the human brain and memory. Similar to an artificial system like SPECTER, the human brain has to make sense out of an overwhelming amount of sensor information delivered by the human senses. Obviously, performing sophisticated reasoning based on such low-level information is impractical due to the sheer amount of data to process. Thus, the human brain is organized in different stages, which successively perform information reduction and abstraction.

On the lowest level, the so-called *sensory registry* is responsible for collecting and short-term buffering of basic perceptions. This includes conscious perceptions as well as unconsciously made ones, like for instance the last few words of a conversation we can hear but are not paying attention to. In the human brain, the sensory registry has two main purposes: On the one hand, perceptions arriving at the sensory registry may trigger reflex actions even before we get conscious about them. This may save valuable time and lower the overall cognitive load. On the other hand, if something unexpected is happening, we can reconsider the situation as a whole by incorporating previously recognized but ignored perceptions that would have been otherwise lost.

Both are properties useful for a system like SPECTER: Service triggers occurring in certain situations and contexts can be seen as reflexes of the system, while a short-term but rich history of perceptions is useful to interpret and understand newly arriving information. For these reasons, the first part of our memory model is organized similar to the human brain's sensory registry stage.

In the next stage of the human's brain, perceptions of the sensory registry are transformed into more abstract experiences and perceptions by memory processes, and are stored in the human's short-term memory. This is reflected within SPECTER by an abstraction process we will explain later in this paper. Because there is a close interaction between the sensory registry and the short-term memory, we pooled both in the first stage of our memory model, called SPECTER's artificial *short-term memory*.

After some time in the human's short-term memory, experiences are transferred to the long-term memory stage where they are linked with previous experiences. That way, general attitudes and preferences are established (often we like or dislike something without exactly knowing why), and experiences are related to similar ones which helps to recall them later (the smell of suntan lotion makes us think about our last holiday).

Once again, both are features relevant for a ubiquitous assistant application like SPECTER. Therefore, SPECTER's *long-term memory* was designed in order to allow similar exploitation of recorded experiences. In the first step, experiences from the short-term memory are transferred into a context log storing the plain observations. Additionally, personally coined references between items of the context log are established in the *personal journal.* This in particular includes but is not limited to assignments of favor (user likes or dislikes an experience) and relevance (an experience was considered to be more or less important by the user). These assignments are made either by the user's direct feedback during recording of the observations (for instance via biosensors), or later during an introspection phase. The context log on the one hand allows to link and recall experiences with respect to certain contexts by temporal correlation. On the other hand it serves together with the personal journal as knowledge source for the learning process, which builds and updates a *user model* capturing the user's general attitudes and preferences. Like the personal journal the content of the user model may be reviewed and refined by the user during the introspection process.

Although our design decisions discussed above have been guided by the structure of the human brain, it is important to note that our goal is *not* to build an exact copy of the human brain. As we want to augment and complement the user's memory, there are also fundamental differences to the human brain. The most important one for instance is, that filtering in our short-term memory is by far less restrictive than in the human brain. In our artificial memory, we are trying to gather and store as much information as possible, even if it does not immediately seem to be relevant. That way, we would be able to perform a more in-depth analysis of experiences when required at a later point. For the same reason, at the moment no memory process like the act of forgetting exists in our model. However, older experiences may be assigned a decreasing relevance in reasoning processes in the course of time.

Implementation Details

In this section we want to give details about the current state of our implementation. As we are reporting about ongoing work, the functionality described above has not been fully implemented yet. Therefore, we will focus on the modeling of perceptions, how we store and access them, and what mechanisms we used to implement memory processes.

As a central part of the SPECTER system is the tight cooperation between the user and the system, information needs to be processed in a format meaningful to both. Therefore, we decided to model perceptions and memory entries as instantiations of ontology concepts, based on the IEEE SUMO and MILO ontologies (cf. [12]) with domain-dependant extensions. The main idea is, that each observation made by a sensor forms a self-contained OWL model derived from the underlying ontology classes.

Inside the memory, these perceptions are stored in so called *RDF stores* (with one exception explained later). An RDF store is a persistent collection of arbitrary RDF models with a flexible interface to query and retrieve a collection of models similar to the RDF Net API (cf. [15]). For each model in the store, additional meta information like the source of the model and a timestamp is added. We implemented these RDF stores using Java and the Jena toolkit (cf. [8]). The most important RDF store in the memory model is the context log, which is responsible for the long-term storage of all models of recognized perceptions. The intuition is, that for every type of observation (determined by its ontological class) a virtual "track" exists in the context log. That way, the context at a given point in time can be reconstructed by taking a snapshot of all tracks active at that time. On the other hand, because model content in an RDF store is indexed, all time points with a certain context constellation can be easily identified which is useful for recalling past situations.

The last component we want to describe is SPECTER's memory processes responsible for the transfer of data between different parts of the memory. To implement these processes, we decided to use the JAM planning system (cf. [7]). Doing so, we can define memory processes on a logical level as control strategies working directly on the OWL models of observations and memory items. Thereby, the planning system's fact base is tightly coupled to the respective RDF stores of the preceding memory components. One exception is SPECTER's artificial sensory registry, which is optimized for high throughput instead of long-term storage and is therefore directly implemented by the fact base of the responsible planning process.

Now that the implementation (as described above) has been completed a few weeks ago, we are starting to experiment with different control strategies. Unfortunately, it is to early to present results today. In general our idea is to use a relatively small set of predefined strategies and learn additional rules over time based on user feedback through machine learning.

TRANSPARENT USER SUPPORT

While the short-term memory and the personal journal serve the purpose to store intermediary data and retrievable episodes, respectively, the *user model* (UM) is meant to represent the user's long-term preferences, interests, and goals. This in-depth knowledge about the user is required to enable the system to provide adaptive support, for example

by proactively presenting relevant information or triggering (Web) services that meet the user's expectations.

In order to react appropriately, the system must be able to recognize classes of situations and associate these with the activities that are beneficial to the user. Such classification models for situations are derived from lower-level features using a variety of machine-learning techniques. Our current implementation uses decision trees and Naive Bayes.

For a truly ubiquitous system like SPECTER that affects the user's daily life, trust is an important issue. Therefore, the transparency of central processes is an indispensable prerequisite for the acceptance of such a system. Only this way can the system make the user build trust into its mastery of her preferences and, thus, increase the user's acceptance of the overall system behavior (see [2]). This particularly applies to all processes dealing with the acquisition of the UM—such as deriving additional features from sensor data or hypotheses about the user's characteristics—as well as those processes actively using this information to steer the system behavior.

In the attempt to find an acceptable tradeoff between powerful user control and the inherent burden of growing complexity, we designed an intuitive interface that allows the user to interfere even with complex machine-learning processes without the need to deal with technical subtleties of feature selection or data encoding (cf. [3]). The central idea of our approach is to combine the system's capability to deal with *statistical* relevance of a situation's features with the user's ability to name *semantically* meaningful concepts that can and should be used to describe her decision making.

Assume the system tries to create a model that classifies situations according to whether or not a certain service should be executed. For instance, in our shopping scenario the system tries to predict whether or not the user should be presented an advertisement of a nearby store.

In this situation the system will propose a candidate decision tree based on information gathered from the context log and the personal journal. The labeling of training instances stems from user feedback given as a reaction to the system's behavior in previous situations. In order to hide the whole complexity of the classification model, the system only presents what concepts were used to describe episodes from the user's history and discriminate between the two types of situations. Communication between user and system is further facilitated as the user is only shown higher-level semantic features taken from a domain ontology and containing human understandable concepts. The user can then remove (semantically) irrelevant features from the system's list and replace them by other, semantically related concepts taken from the same ontology.

Navigation through the semantic neighborhood of a criticized feature is supported by the system using either a graphical or a list-based interface (see Figure 2). Then the system will re-encode the training data (using a number of heuristics hidden from the user) taking into account the user's specification and iterate until the user is satisfied with the result.[1]

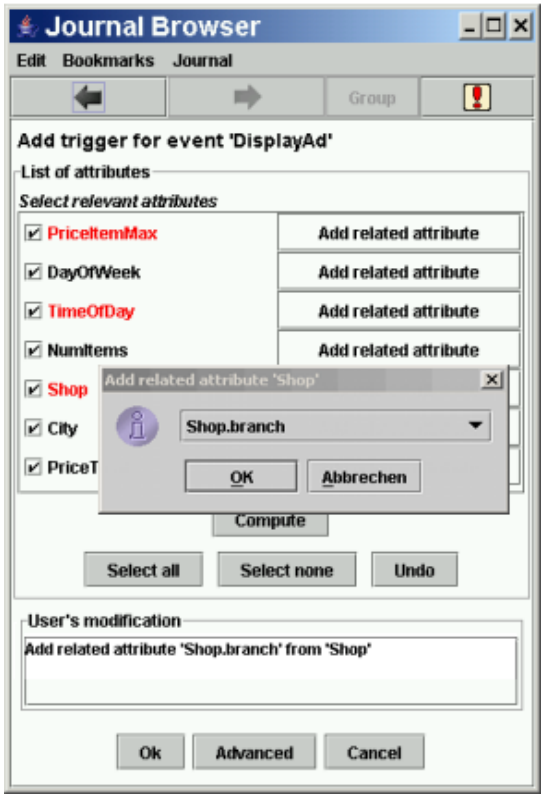


Figure 2: User interface for criticizing and "semantic adjusting" of features selected by the machine learning system due to their statistically relevance.

RELATED WORK

The work described in this contribution is related to several research areas. One central idea is creating an artificial memory, an issue that was the subject of related research, which differed widely in approach and nature of the created memories. In 1994, Lamming and Flynn created a log from sensor input, and pointed out how context information could be applied as a retrieval cue for recalling events in the environment [11]. The permanently growing storage media in mind, Gemmel et al. suggested to digitize the documents created during one's life in order to create a kind of document-centered memory [6]. An example of a product, which has recently appeared in this area is the Nokia LifeBlog software, where data are collected from a camera-equipped cell phone, and are stored in a long-term diary (see *http://www.nokia.com/lifeblog*).

[1] We are currently carrying out a user study to identify the best way to convey the information contained in a decision tree to a naive user.

These projects illustrate the general interest in collecting and filing data, and demonstrate approaches for domains, where rich content is available. However, such input is not necessarily provided by a sensor. Thus, in order to obtain meaningful information from input such as GPS or video, one has to perform an abstraction process in some way. For instance, in [5] clustering of multimedia data is performed to create a diary of situations (e.g., "at the office"). For recognizing basic user states (e.g., "sitting") from acceleration data, in [9] Bayesian classification is employed. An alternative approach is discussed in [14], where objects involved in an activity are mapped to an activity structure mined from the Web.

The memory creation process in SPECTER includes some of the previously mentioned ideas, adapted to the project's specific requirements. These include the collaboration with the user (e.g., to actively collect feedback in response to ambiguous input), and the provision of a mechanism which lets the user add value to the memories beyond archiving.

CONCLUSION

In this paper we sketched the architecture of a system implementing personal, situation-aware, ubiquitous assistance. In order to achieve this goal, the system compiles a kind of memory of observed events comprising aspects of both short-term and long-term memories. The *personal journal* is a kind of episodic memory that can be browsed for interesting events of the past and forms the basis for adaptive user support in a variety of situations.

Machine learning techniques are used to extract relevant patterns from that memory, and thus the user's observed past. These patters allow the system to proactively initiate certain system activities when a particular kind of situation is anticipated or recognized by the system. The collection of these top-level abstractions of the original sensor data forms the core of the *user model* that reflects the user's preferences and expectations in certain classes of situations.

The user largely remains in control over the system behavior and the way it uses her personal data without being forced to engage in lengthy dialogs or deal with complex technical issues. Current work includes the evaluation of certain system aspects w.r.t. usability issues and the integration of various types of sensors providing the low-level input. Future work will be devoted to testing the overall system in complex, mobile scenarios involving a variety of users and services.

ACKNOWLEDGMENTS

Research on SPECTER is being sponsored by the German Federal Ministry for Education and Research (BMB+F under contract 524-40001-01 IW C03.

REFERENCES

1. Amazon Inc. Amazon E-Commerce Service, *http://www.amazon.com/gp/aws/landing.html,* 04/13/05.
2. Bauer, M. Transparent user modeling in SPECTER. *Proceedings of the Seventh International Conference on Work with Computing Systems*, Kuala Lumpur, Malaysia, 2004.
3. Bauer, M., and Baldes, S. An Ontology-Based Interface for Machine Learning. In John Riedl, Anthony Jameson, Daniel Billsus, and Tessa Lau, editors, IUI 2005: International Conference on Intelligent User Interfaces, pages 314–316, New York, 2005. ACM.
4. Brandherm, B., and Schwartz, T. Geo referenced dynamic Bayesian networks for user positioning on mobile systems. *Proceedings of the International Workshop on Location and Context Awareness*, Munich, Germany, 2005.
5. Clarkson, B., Mase, K., and Pentland, A. The Familiar: a living diary and companion. *Proceedings of Computer-Human Interaction*, 2001.
6. Gemmell, J., Bell, G., Lueder, R., Drucker, S., and Wong, C. MyLifeBits: fulfilling the memex vision, 2002.
7. Huber, M. JAM: A BDI-theoretic mobile agent architecture. *Proceedings of the 3rd Conference on Autonomous Agents*, pages 236–243.ACMPress, 1999.
8. JENA – A semantic web framework for Java, *http://jena.sourceforge.net/*, 04/14/05.
9. Kern, N., Schiele, B., and Schmidt, A. Multi-sensor activity context detection for wearable computing. *Proceedings of the European Symposium on Ambient Intelligence*, volume 2875 of LNCS, pages 220–232. LNCS, November 2003.
10. Klatzky, R.L. *Human Memory: Structures and Processes*. W. H. Freeman, San Francisco, 1975.
11. Lamming, M., and Flynn, M. Forget-me-not: intimate computing in support of human memory. *Proceedings of the FRIEND21 Symposium on Next Generation Human Interfaces*, 1994.
12. Niles, I., and Pease, A. Towards a standard upper ontology. *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 2–9. ACM Press, 2001.
13. Obendorf, H., Weinreich, H., and Haß, T. Automatic support for web user studies with SCONE and TEA. *CHI'04: Proceedings of the Conference on Human Factors in Computing Systems*. ACM Press Wien, Austria, 2004.
14. Philipose, M., Fishkin, K.P., Perkowitz, M., Patterson, D., and Haehnel, D. The probabilistic activity toolkit: Towards enabling activity-aware computer interfaces. Technical Report IRS-TR-03-013, Intel Research, Seattle, 2003.
15. World Wide Web Consortium (W3C), RDF Net API, *http://www.w3c.org/Submission/rdf-netapi*, 04/14/05.

Author biographies

Michael Schneider holds a Master of Computer Science from the Saarland University. He works as a researcher at the German Research Center for Artificial Intelligence (DFKI). His interests are ubiquitous computing, mobile systems, and intelligent user interfaces.

Mathias Bauer is a research fellow of the German Research Center for Artificial Intelligence (DFKI). He is interested in user modeling and machine learning/data mining.

Alexander Kröner holds a Ph.D. in Computer Science from the Saarland University. Currently he is working as a senior researcher at the German Research Center for Artificial Intelligence (DFKI). His field of experience comprises the constraint-based creation of multimedia presentations and the application of semantic web technology. In the project SPECTER, his major interest is on the organization and presentation of information stored in the so-called personal journal, a kind of personal user diary.

Proceedings of ECHISE'05, held in Conjunction with PERVASIVE'05, May 11, 2005, Munich, Germany.

MoBe: Context-Aware Mobile Applications on Mobile Devices for Mobile Users

Paolo Coppola[1], Vincenzo Della Mea[1], Luca Di Gaspero[2], Stefano Mizzaro[1], Ivan Scagnetto[1], Andrea Selva[1], Luca Vassena[1], Paolo Zandegiacomo Riziò[1]

[1]Department of Mathematics and Computer Science
[2]Department of Electrical, Management, and Mechanical Engineering
via delle Scienze 206 – University of Udine – I-33100 Udine, Italy
{coppola,dellamea,mizzaro,scagnetto,selva}@dimi.uniud.it, l.digaspero@uniud.it, {lucavax,zandepaolo}@inwind.it
http://www.mobe.it

ABSTRACT

Due to the appearance and widespread diffusion of new mobile devices (PDAs, smartphones etc.), the traditional notion of computing is quickly fading away, giving birth to new paradigms, where concurrent entities, moving from one location to another, exchange data and cooperate towards a common goal. Hence, the scientific community is searching for models, technologies, and architectures in order to suitably describe and guide the implementation of this new computing scenario. It is clear that the notion of context plays a fundamental role, since it influences the computational capabilities of the devices that are in it.

The present work directly addresses this problem proposing MoBe, a novel architecture for sending, in push mode, mobile applications (that we call MoBeLets) to the mobile devices on the basis of the current context the user is in. The latter is determined by both an ad-hoc MoBe infrastructure and data from sensors on the mobile device (or in its surroundings).

Keywords

Mobile devices, context-aware, software architecture.

INTRODUCTION

We envisage a world in which the mobile devices that everybody currently uses (cellular phones, smart phones, PDAs, and so on) constantly and frequently change their functioning mode, automatically adapting their features to the surrounding environment and to the current context of use. For instance, when the user enters a shopping mall, the mobile phone can provide him/her with applications suitable for shopping, i.e., article locator, savings advertiser etc; when entering in a train station, the same device becomes a train timetable able to give information on the right train lane, delays, etc.

How to achieve this goal is not clear. It is well known that current mobile devices can be used as computers, since they have computational and communication capabilities similar to computers of a decade ago. One approach might be to have an operating system continuously monitoring sensors on the mobile device, thus inferring situational information and triggering the right (preloaded) application for the current context. Another approach is to have a Web browser showing to the user context-aware data selected by means of information filtering techniques.

In our opinion both these alternatives suffer from a lack of flexibility and a waste of computational power. We propose a different approach, where servers continuously push software applications to mobile devices, depending on the current context of use. Inspired by the well-known Nicholas Negroponte's "Being Digital" expression, we name our approach *MoBe* (Mobile Being), and the context-aware applications pushed and executed on the mobile device *MoBeLets*.

This is an interdisciplinary work: mobile agent community, context aware computing, software engineering and middleware, interaction with mobile devices applications, information retrieval and filtering, and privacy and security management are all disciplines that are deeply involved in our project.

In this paper we describe our approach and some details of its ongoing implementation, emphasizing how the MoBe architecture efficiently supports a notion of context history.

The paper is structured as follows. In Section "Related Work" we recall the state-of-the-art in the literature for the research fields related to our work. In Section "The Overall Architecture of MoBe" we describe the structure of our model. In particular we give some details about the key submodules dealing with the data sensing and context inference activities, with the personalization issues, and with the problem of filtering, downloading and executing the MoBeLets. Section "Discussion and Open Problems" is devoted to the analysis of several practical issues we found during our first prototype of the MoBe architecture. Moreover, we also explain how the MoBe architecture naturally supports context histories.

RELATED WORK

This is an interdisciplinary work and there are several related fields.

Context-aware computing is more than 10 years old, as it was first discussed in [8]. However, the field seems still in its infancy, as even the core definition of context is still unsatisfying. Some definitions are, like dictionary definitions, rather circular, since they simply define context in term of concepts like "situation", "environment", etc. Some researchers tried to define this concept by means of examples [2,9]; other researchers searched for a more formal definition [2,3,10]; others identified context with location [8] or with location, time, season, etc. [1,7].

An interesting framework for the development of location aware applications is described in [11], where a symbolic location model is used to represent the user's situational context and a map modeling tool links the symbolic information to the corresponding geographical coordinates. The resulting hierarchical structure is encoded in XML and can be accessed through the WWW, without the need of an explicit server infrastructure.

Another related research field concerns mobile agents [12]. Our approach tries to avoid all the resource load that these architectures usually carry with, and to provide a simpler implementation.

Information retrieval, context aware retrieval, just-in-time information retrieval, and information filtering deal with the information overload problem from different facets [6] [5]. Google is starting to provide contextual (actual, localized) services as well (http://www.google.com/lochp). Peer-to-peer networks and wireless networks and technologies are of course involved as well.

THE OVERALL ARCHITECTURE OF MOBE

Figure 1 shows the overall MoBe architecture. The mobile device runs a software module called *MoBeSoul* that is responsible of managing the whole lifecycle of a context-aware application. Let's follow the events that lead to pushing, downloading, and executing a MoBeLet on the mobile device.

Context submodule

The process starts with context data received through:

- *Physical sensors*. Almost all mobile devices are equipped with some form of wireless network technologies (GSM, GPRS, UMTS, Bluetooth, Wi-Fi, Radio Frequency, IrDA, etc.), and can therefore sense if there is a network connection around them (and the strength of the corresponding electromagnetic field). Moreover, the device might be equipped with sensors capable of sensing data about the physical world surrounding the mobile device (e.g., noise, light level, temperature, etc.); also, the device might be able to receive data about its environment (e.g., temperature, etc.) from some surrounding sensors.
- *"Virtual" sensors*. MoBeSoul might receive data from other processes running on user's mobile device, like an agenda, a timer, an alarm clock, and so on.
- *MoBeContext sensors*. MoBeSoul is capable of receiving context information provided by an ad-hoc *MoBe Context Server* (*MCS*). The MCS pushes information about the current context to the users devices, with the aim of providing a more precise and complete context description. MCS might be implemented by a Wi-Fi antenna, an RFID tag sensed by the mobile device, or any other technology. The MCS also broadcasts a Context ID (that, in the case of a Wi-Fi antenna might be the network SSID and its MAC address).
- *Explicit user actions*. The user can explicitly communicate, via the user interface, data about the current context. For instance, he/she might choose a connection/network provider, set the alarm clock, select the silent mode, and so on.
- *Context history representations*. Sequences of contexts traversed by the user in the past can be summarized in some abstract form and used as context data as well, together with the other kinds of context.

All these sensors data are processed by the MoBeSoul *Context* submodule. It is responsible of producing, storing, maintaining, and updating a description of the current context the user is in. The Context submodule starts its inferential activity from *concrete contexts* (i.e., contexts directly corresponding to sensors data). By some inferential mechanism (we are currently devising a mechanism that employs Bayesian Belief Networks) it derives *abstract contexts* (i.e., context which can be processed more conveniently; some of the abstract contexts might be just concrete contexts). The data and the inference are uncertain, and both the concrete contexts and the inferred abstract contexts have a probability measure representing how likely it is that the user is indeed in those contexts. The inferential engine exploits a database containing the history of past contexts and it is tightly integrated with the Personalization submodule (explained later), managing user's preferences, user's current cognitive load, and degree of attention, etc. Concrete and abstract contexts are represented by means of *context descriptors*; the inferred abstract contexts descriptors are stored in a *Current Context Working Memory*, and they survive until the event of exit from that context is inferred.

Examples of current contexts are: the temperature is 20 degrees (with probability 0.9); the time is 12:30:00PM (p = 0.99); the MoBeContext ID is 1234; and so on.

Examples of abstract contexts are: the user is in a shopping mall (p=0.75); the user is in the AirWood bookshop inside the shopping mall in Udine West; the user is in his/her car (p=0.56); the user is driving a car (p=0.8).

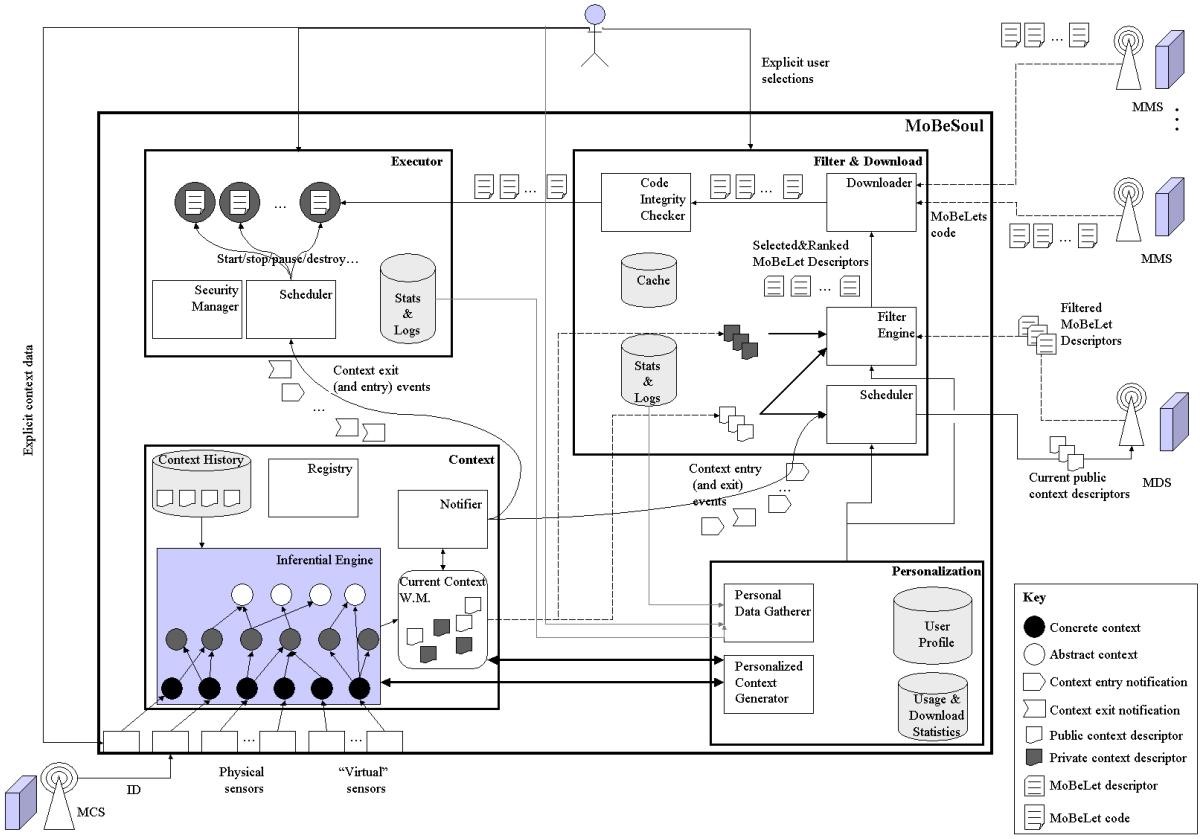


Figure 1. MoBe overall architecture.

Private context histories can be stored and processed only Contexts are divided into a *public* and a *private* part: the former can be distributed to servers and other entities and contains, e.g., user's approximate location, cognitive load, and so on; the latter is kept private inside the MoBeSoul and contains, e.g., user's exact position, credit card information or some other personal data, and so on.

Of course, personal user preferences can change the public/privacy status of each item in a context descriptor. on the user device; public context parts may be sent to external entities able to collect individual context histories and aggregate them for some purpose.

Context submodule does not send autonomously context descriptors to other parts of the system; rather, it keeps a *registry* of interested observers/listeners, which are notified by the *Notifier* when the context entry/exit events happen. After the notification, the observers can decide, using their own criteria, to request the needed context descriptors to the context module.

Personalization submodule

The *Personalization* submodule has two aims:

- The *Personal Data Gatherer* collects data about user's preferences and habits, storing them into two internal databases: the *User Profile* database contains several different kinds of data, like user's demographic information (age, gender, etc.), preferences about real world activities (e.g., restaurants, friends, etc.), habits (working hours, typical trips, etc.), and so on; the *Usage & Download Statistics* database contains data about which MoBeLets have been downloaded and executed in the past, for how much time, which resources have been used, and so on. User's data are collected both automatically (monitoring user's behavior) and manually, by explicit user intervention.
- The *Personalized Context Generator* interacts with the Context submodule, affecting the inference process with the aim of making it more tailored to individual needs. A useful metaphor to understand the interaction between Context and Personalization submodules is to see the Bayesian inferential network inside Context as a graph painted on a sheet of paper, and to imagine the Personalization activity as a transparent sheet of paper on top of it: the Personalization layer is specific to the single user, it has a higher priority and is capable to change the underlying (and more general) context network. The personalization layer can remove (hide) nodes and arcs, change arcs weights (probabilities) either in an absolute way (by specifying a new value) or in a relative way (by increasing or decreasing the underlying weight of a given amount). This also allows to modify in a seamless way the Context network, in order to include unforeseen contexts and inferences even after the system is deployed.

Summarizing, contextual information is derived by the mobile device from physical, virtual, ad-hoc sensors, and user data; the Context and Personalization submodules infer an abstract description of the current context taking into account, besides concrete context data, inference rules, user's preferences (history, user model, …), user's current activities, cognitive load, degree of attention, other devices proximity, etc. A clear separation between context and personalization seems difficult to realize, but has important benefits: independent modification of the Context network, independent usage of well established techniques from both the personalization and context-awareness fields, development of a non-personalized prototype of the MoBeSoul, and so on. However, the relationship between context-awareness and personalization should be carefully studied, since, e.g., context histories might be viewed as a source of personalization data.

Filter and Download submodule

The *Filter and Download* submodule is in charge of selecting which MoBeLets to download and to retrieve their code. It is triggered by notifications of context entry and exit events, received from the Context submodule. The *Scheduler* receives these notifications and, on the basis of its internal criteria, also depending on user's preferences, decides when to request the current public context descriptors to the Context submodule and to forward them to a *MoBe Descriptors Server* (*MDS*). The MDS is in charge of selecting, on the basis of the received context descriptors, those MoBeLets that are more relevant to user's current context.

Since not all the MoBeLets selected on the basis of the public context descriptors will be downloaded (nor executed), the MDS does not handle MoBeLet code, but just *MoBeLets descriptors*. Each descriptor is a simple XML file containing several data about the corresponding MoBeLet: an unique identifier, a textual description, a manifest declaring which resources the MoBeLet will need and use while executing, a download server from which the actual MoBeLet can be downloaded, and so on.

The received MoBeLet descriptors are filtered once again by the *Filter Engine*, using the private context descriptors. As a result of this step, the probability that the user will desire to run each MoBeLet is determined. Then the *Downloader* downloads, on the basis of its own internal criteria, the MoBeLets code, from the *MoBe MoBeLet Server* (*MMS*) specified in the corresponding descriptors. The stream of MoBeLets is then passed to the Executor.

This design allows:

- To encapsulate inside the Scheduler adequate strategies to send to the MDS the public context descriptors, for a more efficient resource usage: the Scheduler might send the context descriptors at each context change, it might collect a certain number of context descriptors (perhaps removing those corresponding to context exit events received meanwhile), it might send context descriptors at fixed time points, and so on.
- To separate public and private context data: only the public data are sent to MDS, but both public and private are used to filter the MoBeLet descriptors received.
- To easily cache both MoBeLet descriptors and code, in order to minimize bandwidth usage.
- To have the user controlling the whole process and to participate in MoBeLets filtering and selection: the user might proactively stop an undesired MoBeLet, or be requested a preference to a resource demanding MoBeLet, and so on. On the other side, the two stage filtering allows a lower cognitive load to the user.

Executor submodule

The last submodule of the pipeline is the *Executor*. Its aim is to run each downloaded MoBeLet inside a sandbox, in order to avoid malicious MoBeLets to use resources against user's will. Each MoBeLet is managed by the *Scheduler*, which is capable of starting, pausing, stopping, and destroying the MoBeLets. The Scheduler is notified of context exit (and entry) events, to stop those MoBeLets that go out of context. Each MoBeLet can register itself with the Registry inside the Context submodule, in order to be directly notified of relevant context change events.

Each MoBeLet that has to use resources outside its sandbox is allowed to do so only through the *Security Manager*, which will deny requests that are incompatible with MoBeLet manifest, prompting the user to confirm more heavy resource usages.

DISCUSSION AND OPEN PROBLEMS

We described an architecture that is still under development; in this section we focus on some open issues.

Scalability issues

MoBe architecture is scalable for what concerns MCS and MS: more servers can be added at will, since each of them does not provide a centralized service. The bottleneck of this architecture is the MDS: in some cases, the MoBeLet descriptors request will be sent to some local server (when the MCS provides a context ID); but in some other cases the MoBeLet descriptors request will be sent to the main MDS server (when the ID can't be provided). In the last case, there is the risk of overloading the main MDS server.

To understand if this is a serious problem, let us try to compare it to nowadays Google statistics. Google receives, and processes almost immediately, about 1000 queries per second. If MoBe will be adopted, we can estimate about 1 billion of MoBe enabled mobile devices, each of which will probably perform, on average, about 1000 context change per day (in daytime, about 50-100 context change per hour; no context change during the night). This would mean a total of 10^{12} context change per day, i.e., $(10^{12}) / (24 \times 60 \times 60) \approx 10^{7}$ ca. context change per second. Not all of them will be sent from MoBeSoul, since the Schedule submodule Filter & Download selects and queues some public context descriptors, but let us be pessimistic and assume that this does not decreases significantly the number of requests to the public server. Let us assume instead that the local

server allow to decrease of another factor of 10, leading to 10^6. This is 1000 higher than today's Google, but it is not so frightening; at worst, we might deploy 1000 MDS around the world, and configure the MoBeSouls so that each of them talks to one of these (e.g., randomly, or statically), thus distributing the load. As a last note on this issue, let us remark that in principle MCS, MDS and MMS can be the same server.

Structured vs. unstructured approach

Turning to more general issues, we see two major trends in current computer science and web technologies. The first one is to provide *structure* in the produced data: in databases, data are stored and retrieved accordingly to well defined schema; XML, HTML, XHTML can instill semantic information in otherwise almost unstructured natural language text; Web services are described on the basis of specific XML formats; Semantic Web is a hot word in the community; and one might go on. Research within the second trend is devoted to empower current algorithms, techniques, and software applications in order to deal with unstructured data: search engines are the second activity of web users (after email); Google GMail fosters an unstructured view of one's own mailboxes; images, sounds, and videos are often searched on the basis of their semantic content, which is hard to encapsulate in a-priori textual descriptions; and so on.

MoBe tries to combine both approaches: a context descriptor is are made of structured data; a MoBeLet descriptors can be mainly made of structured data, provided by the MoBeLet creator, but in principle it is possible to have also unstructured data like, e.g., the comments inserted in the code by the programmer and to exploit state-of-the-art software retrieval and filtering techniques [4].

Application vs. data

Within MoBe, applications are sent around, not just data. Of course, this is a subtle distinction: as every student knows, for a compiler an application is simply data; moreover, looking inside the memory of a computer, one cannot distinguish between bytes representing programs and bytes representing data on which programs run. However, from an abstract/semantic viewpoint, it is perfectly reasonable to distinguish between the two.

Therefore, MoBe approach is different from current mainstream that relies on Web browsers based on HTTP-like protocols (HTTP, WAP, etc.). We believe that this is a shortsighted view: using a well-known metaphor, we might be experiencing the QWERTY of mobile/contextual applications/devices. MoBe is a much more flexible and powerful architecture. Of course, we are aware that it has its own weaknesses: writing software instead of data is more difficult; sending applications might lead to spread malicious MoBeLets (i.e., viruses); privacy issues, handled by distinguishing between public and private context parts, are much more complex, and so on.

Context histories, context, and personalization

MoBe architecture is somehow neutral with respect to context histories, but it takes them into account in a rather natural way. First, the Inferential Mechanism inside the Context submodule infers the abstract context not only on the basis of the current data from the sensors, but also exploiting the context history database. Second, downloaded and executed MoBeLets can be selected not only on the basis of the current context, which in turns depends on the context history, but also exploiting the Statistics & Log databases inside the Filter & Download and Executor submodules.

This is another point in which the aforementioned separation between context and personalization is, although tricky, advantageous, since it can simplify and empower context histories management. Indeed, statistics and logs of MoBeLet usage by a user are rather sensible data; hence, they can be exploited at Personalization (rather than Context) level. On the other side, average statistics on MoBeLets download and usage could be kept on the MoBe Descriptor Server, to provide a more effective filtering by the public context descriptors. Finally, the distinction between context-aware and personalization (and public and private context) is a complex issue deserving further work.

REFERENCES

[1] P.J. Brown, J.D. Bovey and X. Chen. Context-aware applications: From the laboratory to the marketplace. *IEEE Personal Communications* **4**(5): 58–64. 1997.

[2] G. Chen and D. Kotz. A Survey of Context-Aware Mobile Computing Research, 2000

[3] A.K. Dey and G.D. Abowd. *Towards a Better Understanding of context and context-awareness*. TR GIT-GVU-99-22, Georgia Inst. of Tech., College of Computing, 1999. ftp://ftp.cc.gatech.edu/pub/gvu/tr/1999/99-22.pdf

[4] R. Gonzales, K. van der Meer. Standard metadata applied to software retrieval. *J. of Inf. Science*, **30**(4): 300–309, 2004.

[5] G.J.F. Jones, P.J. Brown. Context-aware retrieval for ubiquitous computing environments, In F. Crestani, M. Dunlop, S. Mizzaro (eds.) *Mobile and ubiquitous information access*, vol. 2954 of LNCS: 227–243, Springer-Verlag, 2004.

[6] B.J. Rhodes, P. Maes. Just-in-time information retrieval agents, *IBM Systems J.*, **39**(3–4): 685 – 704, 2000.

[7] N. Ryan, J. Pascoe and D. Morse. Enhanced reality fieldwork: the context-aware archaeological assistant. *Computer Applications and Quantitative Methods in Archaeology*. V. Gaffney, M. van Leusen and S. Exxon (eds.). Oxford (UK), 1998.

[8] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *IEEE Networks* **8**: 22–32, 1994.

[9] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *Proc. of IEEE Workshop on Mobile Computing Systems and Applications*, 85–90, Santa Cruz, CA, 1994.

[10] A. Schmidt, K. Asante Aidoo, A. Takaluoma, U. Tuomela, K. van Laerhoven, and W. van de Velde. Advanced interaction in context. In *Proc. of 1st Int'l Symp. on Handheld and Ubiquitous Computing*, 89–101, Karlsruhe (Germany), 1999.

[11] C. Stahl, D. Heckmann. Using Semantic Web Technology for Ubiquitous Hybrid Location Modeling. In *Workshop notes on Ubiquitous GIS*, in conj. with Geo-Informatics 2004.

[12] M. Wooldridge. *An Introduction to Multiagent Systems.* John Wiley & Sons, 2002.

Author Biographies

Paolo Coppola, PhD, is an assistant professor of computer science at the University of Udine, Italy. His main research interests include the implementation of functional languages, type systems, linear logic, optimal reduction, lambda calculus and computational complexity. He is currently investigating aspects of mobile systems and context-aware computing.

Vincenzo Della Mea, PhD, is an assistant professor of computer engineering at the University of Udine. His research focuses on medical informatics and telemedicine, evaluation of medical systems, hypermedia, eLearning technologies, and mobile devices.

Luca Di Gaspero, PhD, is an assistant professor of computer engineering at the University of Udine. His main research interest concern the investigation of search techniques for scheduling problems. Recently he became interested also in information retrieval and in mobile systems and context-aware computing.

Stefano Mizzaro, PhD, is an assistant professor of computer engineering at the University of Udine. His research activities are mainly in the fields of Web information retrieval, artificial intelligence, scholarly publishing, and mobile devices.

Ivan Scagnetto, PhD, is an assistant professor of computer science at the University of Udine. His research interests are formal methods, computer aided formal reasoning, process algebras and Logical Frameworks. He is currently investigating aspects of mobile systems and context-aware computing.

Andrea Selva holds a M.Sc. in Computer Science from the University of Udine. He is currently a research associate at the same university, working on the implementation of mobile systems.

Luca Vassena is a M.Sc. student in Computer Science at the University of Udine. His master thesis concerns the investigation of context representation techniques for mobile systems.

Paolo Zandegiacomo Riziò holds a M.Sc. in Computer Science from the University of Udine. He is currently a research associate at the same university, working on the implementation of mobile systems.

Proceedings of ECHISE'05, held in Conjunction with PERVASIVE'05, May 11, 2005, Munich, Germany.

The Geographic Context Browser

John Krumm
Microsoft Research
One Microsoft Way
Redmond, WA 98052 USA
+1 425 703 8283
jckrumm@microsoft.com

ABSTRACT
This paper discusses new techniques for enhancing a sequence of (latitude, longitude) points by tagging them with nearby points of interest and associated Web pages. We present a browser that lets a user explore the enhanced tracks by clicking on a map and filtering over aspects of context.

Keywords
Context, geographic information systems, location

INTRODUCTION
Viewing a track of (latitude, longitude) points on a map is an interesting way to relive a trip. The map helps trigger memories about locations and events. Such tracks will become more common as more devices become aware of their own location, such as GPS-equipped cell phones.

While a simple track display is useful, it is possible to extract more context from a series of (latitude, longitude) points. For instance, a (latitude, longitude) can be used to find nearby points of interest (POI) from a database. This paper first describes a few simple methods to extract richer context from a track. Then it shows how the extracted context can be used to look through a track to give a richer, more informative browsing experience, including lists of nearby points of interest and associated Web pages.

LOCATION TRACKS
It is easy to create a location track by carrying a GPS receiver, and even inexpensive GPSs have recording capabilities. Other researchers have investigated the use of such tracks. In Project Lachesis[1], Hariharan and Toyama analyze location tracks to find "stays" and "destinations" and then build probabilistic models of a user's location behavior. Patterson *et al.*[2] use GPS tracks to help infer a subject's mode of transportation – walking, driving, or riding a bus. While these projects use location tracks to compute higher level features of the user's behavior, our work is aimed at enhancing the tracks to help create a richer diary of a user's travels.

CONTEXT ENHANCEMENT
A track is a series of (latitude, longitude) points, which by themselves are not very meaningful to a typical user. However, there are relatively simple techniques to derive more meaningful information about a track. We have augmented our tracks with a Kalman filter to estimate speed, a POI lookup to find interesting nearby places, and a Web search to find Web pages associated with the POIs. This extra information is stored with the tracks and used as input to our context browser.

Kalman Filter
Speed is a valuable feature for a location track, because it can be used to filter salient subsections of the track. For instance, a speed of zero (or near zero) means the subject was lingering at a certain place. Other speed ranges are indicative of walking, bicycling, or riding in a car, train, bus, or airplane.

The Kalman filter[3] is a well-established technique for statistical estimation based on noisy data. It is especially suited to noisy time-indexed data such as GPS location tracks. In our formulation, the Kalman filter computes smoothed estimates of location and velocity, balancing the

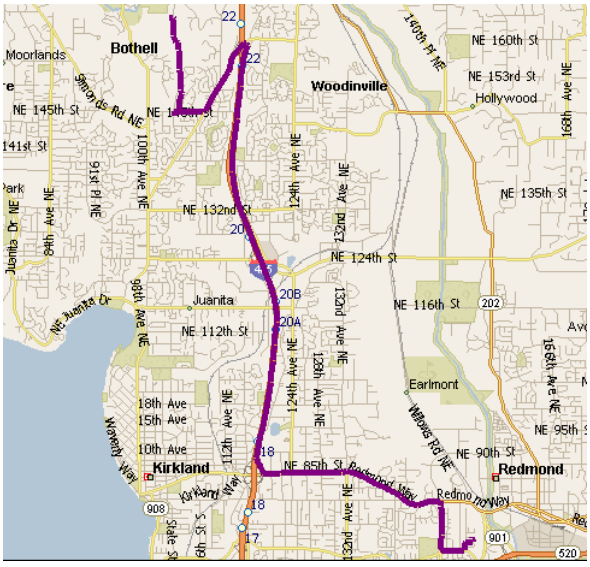


Figure 1: The dark segment starting at the lower right shows a sample location track based on GPS. Our goal is to augment this track with local context information.

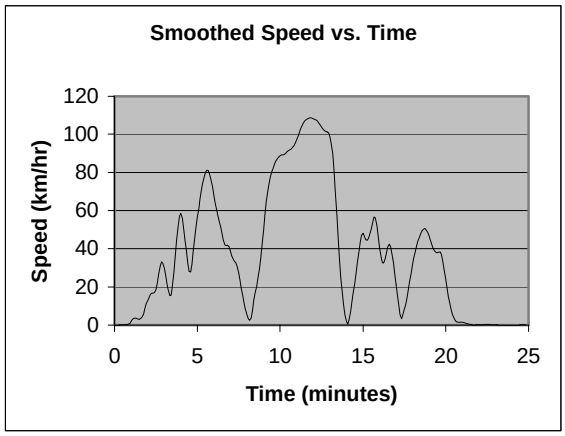


Figure 2: Our Kalman filter computes a smoothed speed estimate for every point in the GPS track. This shows speeds computed for the drive in Figure 1, which had a highway in the middle.

expected standard deviation in GPS location measurements (10 m) and the expected standard deviation in accelerations (0.1 m/s^2). Because the GPS gives only location, using a Kalman filter is an effective way to give realistic speed estimates.

After applying the Kalman filter, we store a smoothed speed estimate for every point of the track. For the drive shown in the map in Figure 1, the computed speeds are shown in Figure 2.

Points of Interest

Points of interest are places like businesses, schools, and cities, normally located with (latitude, longitude). There are POI databases available, some specialized to certain types of POIs. In our project, we use the MapPoint® Web Service, which provides a list of businesses in North America. The businesses are categorized by their North American Industrial Classification System (NAICS) codes. NAICS is a hierarchical classification. For example NAICS code 72 is the category for "Accommodation and Food Services", 722 is the subcategory for "Food Services and Drinking Places", and 7221 is the subcategory for "Full-Service Restaurants". MapPoint® provides an API to take a (latitude, longitude), a search radius, and a NAICS category and return a list of businesses nearby, including the business' name, address, and phone number.

There are 2341 NAICS categories available from. MapPoint®, including exotic types like "Resin, Synthetic Rubber, and Artificial Synthetic Fibers and Filaments Manufacturing" and "Guided Missile and Space Vehicle Manufacturing". We use a subset of NAICS categories, shown in Table 1, to find a list of POIs within 100 meters of each point of our tracks. These POIs are stored for use in our context browser. The POIs found for a particular point are listed in the screen shot in Figure 4.

Web Pages

There are very often Web pages associated with businesses. Unfortunately, these pages are not normally indexed by location, as POIs are. However, we can still find appropriate pages via a Web search using terms from the POIs. For each POI we find, we create a search term consisting of the POI's name and telephone number and submit this to MSN® Search. An example search term shows the different formats we allow for the telephone number:

```
"TRES HERMANOS" AND ("(425) 827-4422" OR "425-827-4422" OR "425 827-4422" OR "425 827 4422" OR "425.827.4422")
```

The first uniform research locator (URL) found for this term gives a review of this Mexican restaurant, shown in Figure 3. The URLs of the found pages are stored along with the POIs and Kalman filter results.

NAICS	Name	NAICS	Name	NAICS	Name
111	crop production	6111	elementary and secondary schools	61161	fine arts schools
112	animal production	6112	junior colleges	71131	promoters of performing arts, sports, and similar events with facilities
442	furniture and home furnishing stores	6113	colleges, universities, and professional schools	71211	museums
443	electronics and appliance stores	6115	technical and trade schools	71213	zoos and botanical gardens
445	food and beverage stores	6211	offices of physicians	71219	nature parks and other similar institutions
447	gasoline stations	6212	offices of dentists	71392	skiing facilities
448	clothing and clothing accessories stores	7111	performing arts companies	71394	fitness and recreational sports centers
452	general merchandise stores	7132	gambling industries	81211	hair, nail, and skin care services
481	air transportation	7211	traveler accommodation	81221	funeral homes and funeral services
482	rail transportation	7221	full-service restaurants	81222	cemeteries and crematories
483	water transportation	7222	limited-service eating places	485112	commuter rail systems
491	postal service	7224	drinking places (alcoholic beverages)	485113	bus and other motor vehicle transit systems
492	couriers and messengers	8131	religious organizations	512131	motion picture theaters (except drive-ins)
622	hospitals	31212	breweries	512132	drive-in motion picture theaters
4411	automobile dealers	31213	wineries	532111	passenger car rental
4441	building material and supplies dealers	44611	pharmacies and drug stores	561439	other business service centers (including copy shops)
4531	florists	45111	sporting goods stores	611620	sports and recreation instruction
4532	office supplies, stationery, and gift stores	45112	hobby, toy, and game stores	711211	sports teams and clubs
4871	scenic and sightseeing transportation, land	45113	sewing, needlework, and piece goods stores	711212	racetracks
4872	scenic and sightseeing transportation, water	45114	musical instrument and supplies stores	713990	all other amusement and recreation industries
4881	support activities for air transportation	45121	book stores and news dealers	721211	RV (recreational vehicle) parks and campgrounds
5112	software publishers	52212	savings institutions	721214	recreational and vacation camps (except campgrounds)
5312	offices of real estate agents and brokers	52213	credit unions	811192	car washes

Table 1: This is the list of types of points of interest we look for around every (latitude, longitude) point. These are a subset of the North American Industrial Classification System (NAICS) categories.

CONTEXT BROWSER

Our context browser is designed to let a user explore a GPS track. It takes as input the enhanced tracks described in the previous section, including the Kalman-filtered speed, POIs, and URLs for each track point. A screen show of our context browser is shown in Figure 4. The upper left shows a clickable map with each track point marked with a disk.

Each track point is clickable, which brings up a list of found POIs and URLs in the upper right panel. Using this capability, a user can click on various points and immediately see a list of what was nearby and, if available, Web pages associated with that location. Clicking on a URL launches a separate Web browser displaying that URL.

The lower left of the UI is devoted to filtering track points. This allows the user to find which track points meet certain criteria in order to target certain subsections of the trip. One of these criteria is a threshold on speed. Invoking this filter assigns gray to all track points below the threshold and black to those above. This is useful for finding, say, parts of the track where the user lingered (zero speed) or where the user was moving at highway speeds. The low- or zero-speed points are especially interesting, because they indicate where the user stopped, perhaps at one of the POIs in the list.

The other available filter selects those track points that are near a selected type of POI as indicated by a NAICS category. The speed and POI filters can be independently enabled and disabled. Taken together, they allow a user to find track points that, for instance, show where the user stopped (zero speed) near a gas station with a convenience store (NAICS code 447110).

CONCLUSION

Using relatively simple means, our system enhances tracks of raw (latitude, longitude) to include speed, POIs, and Web page URLs. Our context browser lets a user explore the track points and their associated context by choosing points on a map and by filtering the context based on speed and type of POI. This is a convenient way to explore tracks and to trigger memories of interesting points along the way.

In the future, the geographic context browser could be enhanced with other sources of information. For instance, adding historical weather data would allow a user to query on weather conditions as a trigger to find parts of a trip, *e.g.* "I remember we were near a movie theater and it was raining." Another interesting set of data to tap is digital photos. Since the GPS creates time stamps for all the (latitude, longitude) readings, it would be easy to find relevant photos from the trip if they were stored on the same computer used to run the context browser. There are also digital photos available via the Web that come with (latitude, longitude) stamps that could be displayed[4].

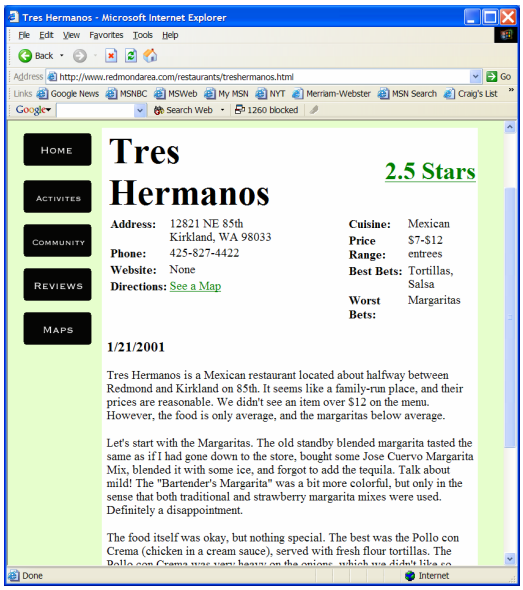


Figure 3: This is one of the web pages found for a track point on the street outside this Mexican restaurant.

REFERENCES

1. Hariharan, R. and K. Toyoma. *Project Lachesis: Parsing and Modeling Location Histories*. in *Third International Conference on GIScience*. 2004. Adelphi, MD, USA.
2. Patterson, D.J., et al. *Inferring High-Level Behavior from Low-Level Sensors*. in *Fifth International Conference on Ubiquitous Computing*. 2003. Seattle, WA, USA.
3. Gelb, A., *Applied Optimal Estimation*. 1974: MIT Press.
4. Toyama, K., et al. *World Wide Media Exchange*. in *http://wwmx.org/*.

BIOGRAPHY

John Krumm graduated from Carnegie Mellon University in 1993 with a PhD in robotics and a thesis on texture analysis in images. He worked at the Robotics Center of Sandia National Laboratories in Albuquerque, New Mexico for the next four years. His main projects there were computer vision for object recognition for robots and occupant detection for cars. Since 1997 he has been a researcher at Microsoft Research in Redmond, Washington, USA concentrating on location tracking of people and devices and on ways of using location data to benefit the user.

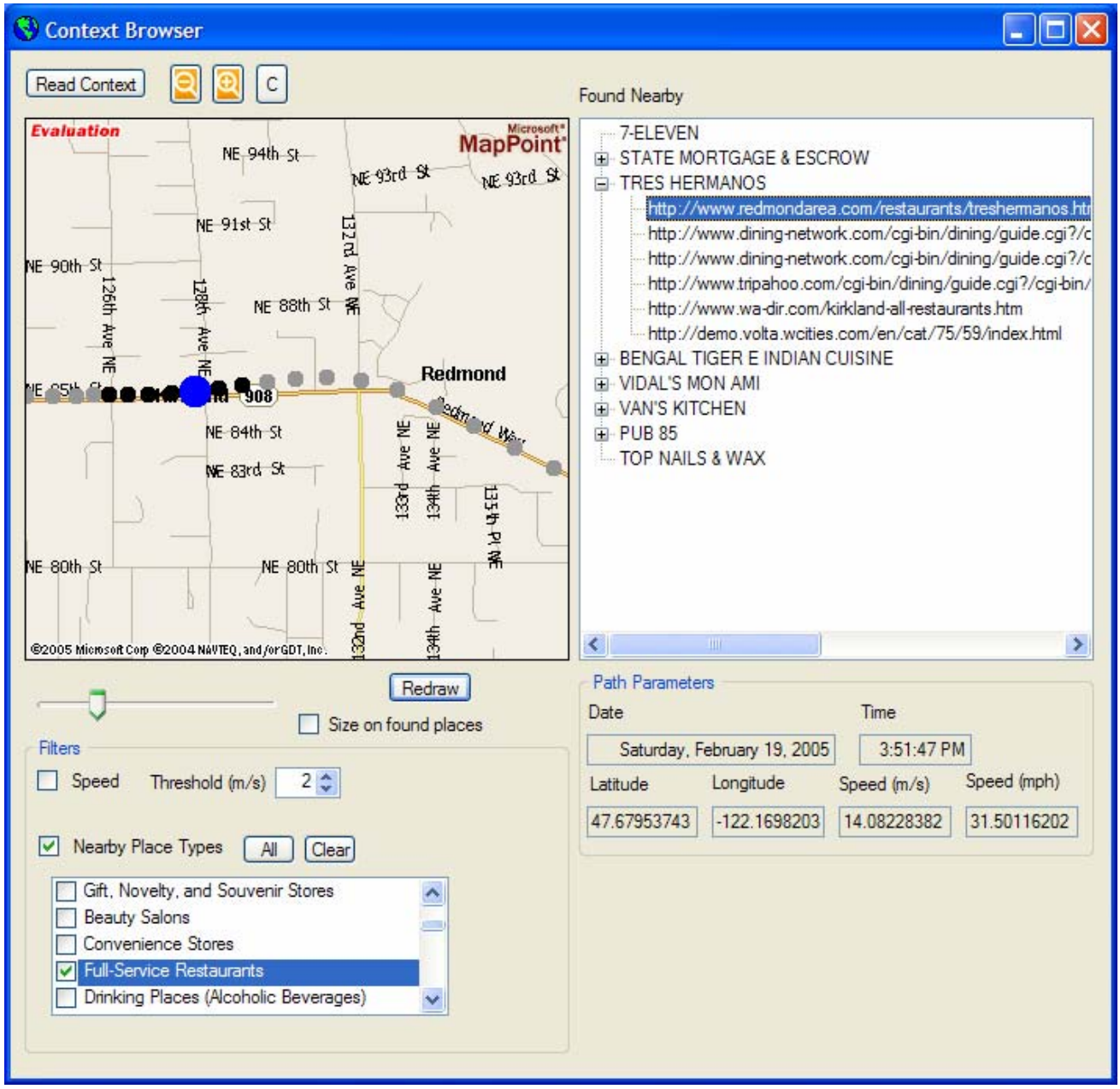


Figure 4: The context browser shows a series of (latitude, longitude) points on a map. The selected point is shown larger than the others. Black dots are locations that are near at least one "Full-Service Restaurant", according to the filter parameters set up in the lower left panel, while gray dots are not. The panel on the upper right shows the points of interest found for the selected point and associated Web page URLs. Clicking on a URL brings up the corresponding Web page. The panel of path parameters on the lower right shows the date, time, location, and speed.

Sharing photos and recommendations in the city streets

Marek Bell, Matthew Chalmers, Barry Brown, Ian MacColl, Malcolm Hall, Paul Rudman
Computing Science, University of Glasgow, Glasgow, UK, G12 8RZ
{marek, matthew, barry, ianm, mh, rudmanp}@dcs.gla.ac.uk

ABSTRACT
Sharing events with others is an important part of many enjoyable experiences. While most existing co-presence systems focus on work tasks, in this paper we describe a lightweight mobile system designed for sharing leisure. This system allows city visitors to share their experiences with others both far and near, through tablet computers which share photographs, voice and locations. A collaborative filtering algorithm uses historical data of previous visits to recommend photos, web pages and places to visitors. In an extensive user trial we explored how these resources were used to collaborate around a physical place.

INTRODUCTION
Co-presence, collaboration and shared experiences between distant individuals are long-standing goals of collaborative systems research [12]. The many limitations of current collaborative technologies, such as telephones and video conferencing, have prompted researchers to explore new ways of sharing space and objects at a distance. Techniques such as moving cameras [17], laser pointers [18], multiple screens [13] and mobile robots [20] have all been used to support shared interactions, at a distance, around physical objects. Previous systems require considerable setup and configuration, and are usually designed for use in stable office or work settings.

We have taken a more lightweight approach to sharing space at a distance between mobile users. Building on our ethnographic studies [2], previous experiments [4], and conceptual work [6], the *George Square* system uses a small, portable tablet PC to allow a mobile visitor to explore a city while sharing their voice, location, photographs and web pages with others. The tablet is connected via the Internet to other users running the same software, who may either be co-present or in different parts of the city. The software can also be run on a standard PC, supporting co–visiting while at home or in a café. The system provides four key resources for sharing the visit. First, users' locations are tracked using GPS and displayed on a map, with non-mobile users able to move an equivalent avatar around by clicking on a map. This supports a shared sense of context in terms of location. Second, users can share photographs taken from an attached camera. Third, the system uses voice-over-IP to support talk and interaction. Lastly, users' ongoing behaviour is recorded and compared to others' past behaviour, to produce a focused set of recommendations of places, web pages and photos displayed on the map. These resources were designed to support synchronous collaboration involving both the online and physical aspects of a city, as well as asynchronous collaboration that exploits the logs of recorded activity, thus creating a shared experience between visitors.

In more general terms, our work is intended to look beyond an individual's use of information by him– or herself, toward *collaborative ubicomp:* mobile or embedded systems that can support users collaborating with others both co-present and distant. Unlike desktop collaborative systems, in mobile systems the specific *place* that users are in can play a significantly larger and more dynamic role in collaboration. However, this information needs to be woven into support for interaction around online media, such as web pages and photographs.

The 'George Square' system extends our earlier work inside museums [4] in a number of ways. The system explores how collaborative ubicomp can work in the city streets rather than one confined location. In this less constrained setting, content is much harder to produce as there are substantially more items for which content must be authored and the set of items is not under our, or any one person's, control. The system therefore makes use of the existing digital information that is available about places, such as maps and web pages already online, as well as allowing users to create their own content. In either case, logs of the use of this heterogeneous mix of information are used as a resource for ongoing activity. In part, this logging is conducted to support visitors' activities before and after their visit. Our observational studies of city visitors emphasised that the visit itself is only one part of a visitor's experience; the 'pre-visit' and 'post-visit' have an important role for both planning and sharing. Our design therefore supports users in planning their visit in advance, and in reviewing their visit afterwards.

Lastly, the growth in community web sites that discuss and share experiences of different places (e.g. trekshare.com) underlined the importance of using others' experiences in shaping the visits. To do this we have experimented with using recommender algorithms to shape what information is

presented to users, taking advantage of the logs from previous visits and 'pre-visits' to assist the current visit. In conjunction with the other features of the system, the use of this past information allows us to develop further the concept of co-visiting, in the form of a lightweight mobile system that can be run almost anywhere with the minimum of configuration and setup.

PREVIOUS WORK

City visiting has been a popular area for mobile information systems, in particular [7], and other PDA based systems [1, 8, 25]. Indeed, as mobile phones and other portable devices become more advanced, tourism seems to be an obvious application area. A number of phone operators have already released city guides for easy viewing that are targeted and customised for mobile phones (for example, http://www.lonelyplanet.com/mobile/). However, these and other commercial technologies have had only limited success. Generally, they are based around a 'walk–up, pop–up' model where information, such as text and pre–recorded speech, is pushed at a user based on his or her current location. This type of model can often seem static and leave the user feeling that the system is not greatly interactive – that they have little input or control and that they are very much working in an isolated environment. There has been little explicit support for *collaboration* between visitors.

One notable exception in this regard was *Sotto Voce*, which allowed museum visitors to share a spoken commentary as they visited a historic house [25]. A small number of mobile systems designed for entertainment and games also specifically address collaboration. *Can You See Me Now*, for example, was a performance that employed a game format. It incorporated multiple players using wireless–enabled PDAs on city streets, who were in turn connected to online players via the Internet [9]. A recent commercial mobile game that relies on collaboration is Newt Games' *Mogi* (www.mogimogi.com), which involved finding and trading objects in city streets.

Similarly, while recommendation systems usually generate recommendations by combining records of several people's past activity, collaboration has seldom been a central focus. PolyLens [19] was one recommender which worked for groups, in that it allowed two or more people to combine their movie rating profiles into one, and then create one recommendation list from this. Also, recommenders rarely use a broad set of contextual features, although the Jimminy system [21] was one temporally–specific single–user recommender that used explicitly–entered textual notes, and the names of locations and people, as contextual features to base recommendations on.

SYSTEM OVERVIEW

In the George Square system (Figure 1), each tourist can visit the physical city much as they would in a normal city visit. On each tablet PC, the visitor's location is tracked using a GPS and shown (1) on a map of the city. Maps are automatically downloaded over the Internet from a map server, allowing the system to be run anywhere. As an alternative to specifying location via GPS, visitors can select a 'manual position' mode, and then click on the map to specify their position.

As a visitor moves around the square, he or she can take photographs of attractions using an attached camera. The pictures are geo-referenced and shown on all users' maps at the location where the picture was taken (2). These pictures are also shown on a shared 'filmstrip' view, alongside buttons to control the map's zoom level, briefly highlight a position on the map, change positioning mode and take a photo.

User context and activity is logged in a database, recording the attractions in the square each user encountered, web pages browsed and photographs taken. This historical information is run through the Recer collaborative filtering algorithm [5] to find attractions and web pages (3) accessed by previous visitors in similar contexts. Pictures taken by visitors in similar contexts are also recommended (4). These recommendations are displayed on each user's map, and in a legend below each map (5). In order to support sharing and discussion, one sees others' recommendations 'ghosted' on one's own map, and sees others' recommendation lists alongside one's own (also ghosted for easy distinction). Map icons for web pages and photos can be clicked to view the related content in detail. Lastly, a voice–over–IP subsystem allows visitors to talk as they visit together.

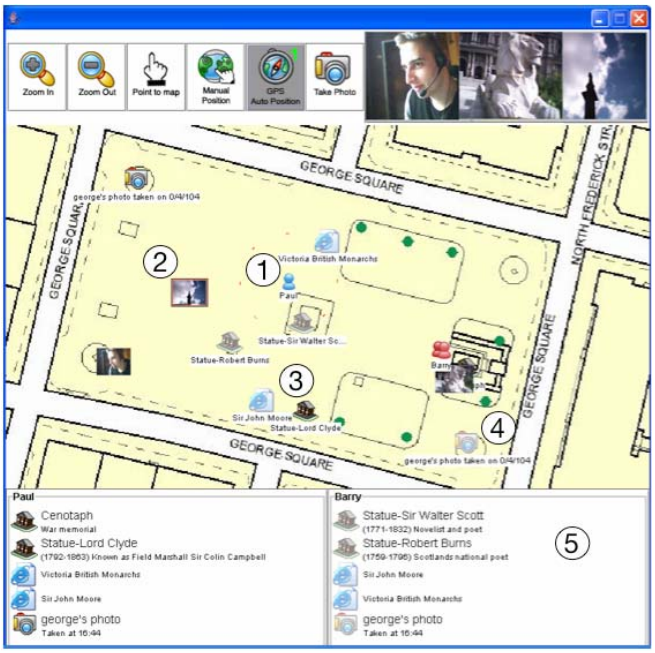


Figure 1: 'George Square' system, showing map that displays each user's location (1), thumbnail photos (2), recommended locations, web pages (3) and photos (4), and each users' recommendation list (5).

In use, the system supports a range of different scenarios. Firstly, it can support two users collaboratively co–visiting an area of a city, taking photographs and browsing web pages about that area. Secondly, it can support users physically present at the location collaboratively co–visiting with other users distant from the where they are, via the

Internet. Thirdly, users who are all distant from the area but interacting via the Internet can use the system to share a purely online visit. The latter scenario is important as this is often 'pre-visiting' in which people explore photographs, web pages and attractions that are of interest before they actually arrive at a city. Vital historical data that can feed into their later activity of the actual visit may be recorded at this pre-visit stage.

In a complementary way, we support post–visit activity. The database log generated from earlier visiting is used to generate a web page: a travel weblog [2]. One can browse the web pages generated from one's visit, viewing a temporally–ordered list of all the pictures, web pages and places that one has visited, and explore a map—based on the one used during the visit. This summarises one's visit in a spatial presentation (the post-visit 'web-log' is discussed in more detail in future papers).

Our use of past activity to build up content in the form of webpages and photographs gives the system considerable flexibility. It can be run in a new city with the minimum of reconfiguration – content does not need to be produced, as it will automatically accumulate from usage of the system. Furthermore, if the system is continually run by waves of visitors then the content will always remain relatively up-to-date as users continue to generate new logs.

The implementation challenges for George Square were typical of other collaborative mobile systems, in that we needed a mix of devices that could work together as peers without relying on access to a central server. We also wanted our system to be dynamic, supporting users and devices joining and leaving at any time.

The hardware of our system consists of a lightweight Tablet PC with attached compact flash GPS unit and a USB 'stalk' camera. Headphones and microphone were plugged into the unit, and the built in WiFi was used for communications. In our trials, a temporary wireless network was bridged to a publicly available WiFi 'hotspot' to provide Internet access. This allowed users to browse and search the web, and to follow links to information provided by our system.

For our software we expanded on previous work with the EQUIP distributed tuple space systems [14], middleware which supports a peer-to-peer communication model between networks of sensors and output devices. EQUIP is used to send data both between the different devices, and system components. Tuple space events are used both for data sharing between components on the same system and network communication to components on other systems, supporting the flexible combination of system components. By using a peer–to–peer architecture, each component can also be used without reliance on a central server. The event–based architecture allows devices and users to leave or join at any time, with dynamic reconfiguration. Events describing user activity and sensor readings are recorded by logging components. These logging components also continually run algorithms comparing recent activity with historical logs, to create recommendations.

USER TRIAL

We ran an extensive user trial of the George Square system in the city streets of Glasgow. In evaluating the system we were sensitive to how it could support enjoyable interactions around place, rather than an optimal, yet potentially sterile, experience. Our focus was thus on the lessons we could learn for designing for enjoyment, as much as evaluating how well our specific system performed. Other papers (under review) report on more general details of interaction with George Square, but here we summarise results related to the use of logged information and recommendations.

We ran a trial with 20 participants, in pairs of two, recruited as pairs of friends. We chose a mix of locals (10) and visitors (10) to the city, recruiting participants through the city's tourist information centre, language schools and our university. Ages ranged from 19 to 35, with 13 female and 7 male participants. Participants were paid for their time at the end of the visit. Each trial lasted between 35 and 60 minutes, with a post-trial debriefing of 10 minutes.

Figure 2: A co-visit with one user physically in the George Square using a tablet PC and one indoors visitor using a laptop to share the visit.

Each pair of users was taken to George Square, an open city square (125 meters by 90 meters) in the centre of Glasgow. This square is a focus for tourists in the city, has a number of statues, monuments and gardens in it, and is surrounded by several major civic buildings. One user was taken to an indoor venue on the corner of the square (the *indoor visitor*), and one visitor was taken out to the square itself (the *outdoor visitor***)**. The outdoor visitor was given the tablet computer as described previously, while the indoor visitor sat at a conventional laptop PC, equipped with a USB camera (Figure 2).

The scenario we gave for the trial was of two friends sharing a visit to George Square, communicating via the system. For the first half of the trial, participants were asked to freely explore the square learning how to use the system. For the second half of the trial, users were given a set of tasks to carry out. This included tasks such as sharing a photograph of the square, and finding out the height of the statue in the centre of the square.

A range of data from each trial was collected: video tapes of both the indoor and outdoor visitors, audio recording of the participants' communication, and log data of the system and users' behaviour. For analysis we combined the shared audio channel and the video images of into a single video stream. From the logs, we generated a 'playback' of the system as seen by the trial participants, and this was superimposed onto the video stream. We also analysed transcripts of the post-trial debriefings, and our general observations of the use of the system.

We were interested in exploring how the system was used, to inform our future designs. Accordingly, we chose a technique known as *interactional analysis* [15], based on paying close attention to the details of how users interact with each other and with technology, usually through the analysis of video. We paid special attention to where the participants used the resources provided by the system, such as location awareness. Having a visualisation of the system's behaviour allowed us to better interpret users' reactions to events. In particular, situations where participants were confused revealed where the system could be improved to better support collaboration or understanding.

In use, the system presented a novel yet enjoyable experience for trial participants, with all participants exchanging photographs, and using their location and recommendations in their interactions around the square. While exactly the same software was used for both indoor and outdoor participants, differences in the visitor's situation resulted in different capabilities for each user. The indoor visitor used a laptop with a larger screen, keyboard and mouse. He or she could type URLs and interact with multiple web pages more easily. However, this user was stationary whilst the outdoor user, through their presence in the square itself, could move around to different statues and attractions, taking photographs of statues and of other events that happened out in the square. These differences in situation led to clear patterns of use and division of labor in the trial. The indoor user would search the web for information about particular statues, whereas the outdoor user would take pictures and relay information about the different statues and their plaques. As one of our outdoor participants put it: "if you can't type, you can't surf the web". However, some web pages were browsed by the outdoor user, since the recommendation system allowed browsing of recommended web pages without having to type in URLs or search terms. These results were confirmed by our analysis of the videos.

The system offered a range of different resources that visitors could use to share the visit: location (displayed on a map), voice, photographs, recommendations and web pages. These different resources supported collaboration between visitors in different ways, but the map proved to be a focal point of collaboration for both the indoor and the outdoor visitor. The indoor visitors made use of the outdoor visitors' location to access the local context of the outdoor visitor, e.g:

```
In: Take a picture of the Robert Burns
    statue---> It's right next to you.
```

Of all the resources provided by the system, the voice connection proved to be the most valuable for creating a sense of shared experience. Through their talk, users continually managed their shared experience, talking about what they were doing, what they had done and what they were going to do. As emphasised in similar studies [9], voice is an essential tool for repairing misunderstandings.

The recommendations of *web pages* acted as an effective way of displaying and linking together the online content available about places, with the place itself. Although we 'bootstrapped' the recommender system by browsing web pages in appropriate places, the system also recommended pages that had been browsed by users during the trials. One early trial participant browsed the 'wikipedia' pages about William Gladstone, which were then recommended to later trial participants who went to the statue of Gladstone. Recommended web pages, positioned on the map, acted as geographical 'bookmarks' in the square being visited, taken from other people's web browsing. These recommendations proved particularly useful to the outdoor visitors, since they could view these recommended web pages by clicking on them, without having to navigate the web.

Along with webpages, our system also recommended sets of places. Places' labels provided the names of different statues in the square, as well as those of buildings on the edge of the square. However, rather than only acting as recommendations of where to go next, these labels acted as labels 'seen in common', which could be used when talking about different parts of the square in sociable or functional ways. The indoor user, for example, could ask the outdoor user to go to a particular attraction by using its name. At times, this conflicted with recommendations' role as suggestions of where to go or what to read next. As a visitor got close to a recommended place, that label disappeared because, from an information–seeking point of view, there was no longer any need to suggest it. However, from a conversational point of view, the shared label for that place was then unavailable as a resource, causing disruption.

DESIGN IMPLICATIONS

One problem with electronic maps, and visualisations more generally, is the need to keep the display clear from irrelevant details. As each visitor in our trial navigated the square, his or her recent behaviour was used to filter the items displayed. Current behaviour was used by our recommendation algorithm to filter and select items from historical data of use. The use of these labels as conversational resources by users suggests how recommenders can be used to filter information displayed on maps in a contextually appropriate manner.

In addition, our recommender also made use of historical data to weave together online information with physical places. Photographs taken, and web pages browsed, by users were stored as an archive of information about the locations the system was used. These associations were not

pre-authored but rather evolved with users' behaviour. This exemplifies how historical data can be a resource for connecting online data with different places.

Both these applications show the value of using recommender algorithms to support collaboration. While tensions exist between single user and collaborative use, we would argue that recommenders and other information seeking tools can be powerful used to support new forms of collaboration.

CONCLUSION

This paper has presented the George Square co-visiting system. The main goal of this system was to support geo–spatial collaboration around a place as well as the information about that place, with a particular focus on support for leisure. The system supports city visitors sharing their visit with those at a distance. It provides resources for sharing voice, photos, location and web pages. A trial of the system uncovered how, through the different resources the system provided, visitors could accomplish a shared visit. In particular, users brought together their shared location, voice, photographs and recommendations to co-ordinate and enjoy a visit.

Ubicomp technology offers the possibility of access to large bodies of information on distant servers and stores, through information-seeking tools such as search engines and recommenders. As well as access to distant information we have shown how it can provide access to distant people and past activity. Collaborative ubicomp can integrate interaction with the local context with the social context of collaborators far away, and historical context in terms of contributors from the past. We suggest that there is rich potential in combining information from near and far, from the past and the present, and from the wide range of tools and media that collaborative ubicomp employs. This paper shows how we can design to support interaction that weaves these apparently disparate places, times, and media into a coherent, manageable and even pleasurable whole.

REFERENCES

[1] Abowd, G.D., Atkeson, C.G., et al. Cyberguide: A Mobile Context-Aware Tour Guide. *ACM Wireless Networks*, 3(5), 1997, 421-433.

[2] Brown, B. and Chalmers, M. Tourism and Mobile Technology. *Proc. ECSCW 2003*, Kluwer, 335-355.

[3] Brown, B. and Laurier, E. Maps & Journeying: An Ethnomethodological Approach. To appear in *Cartographica*.

[4] Brown, B., MacColl, I., et al. Lessons from the Lighthouse: Collaboration in a Shared Mixed Reality System. *Proc. CHI 2003*, ACM Press, 577-585.

[5] Chalmers, M. When Cookies Aren't Enough: Tracking and Enriching Web Activity with Recer. In: *Preferred Placement: Knowledge Politics on the Web*, Rogers, R. (ed.), Jan van Eyck Academie, 2000, 99-102.

[6] Chalmers, M. and Galani, A. Seamful Interweaving: Heterogeneity in the Theory and Design of Interactive Systems. *Proc. DIS 2004*, ACM Press, 243–252.

[7] Cheverst, K., Davies, N., et al. Developing a Context-Aware Electronic Tourist Guide: Some Issues and Experiences. *Proc. CHI 2000*, ACM Press, 17–24.

[8] Fesenmaier, D., Klein, S., et al. (eds.). *Information and Communication Technologies in Tourism*. Springer, 2000.

[9] Flintham, M., Anastasi, R., et al. Where On-Line Meets On-the-Streets: Experiences with Mobile Mixed Reality Games. *Proc. CHI 2003*, ACM Press, 569-576.

[10] Frohlich, D.M., Kuchinsky, A., et al. Requirements for Photoware. *Proc. CSCW 2002*, ACM Press, 166–175.

[11] Galani, A. and Chalmers, M. Production of Pace as Collaborative Activity. *Extended Abstracts of CHI 2004*, ACM Press, 1417–1420.

[12] Gaver, W. The Affordances of Media Spaces for Collaboration. *Proc. CSCW 1992*, ACM Press, 17-24.

[13] Gaver, W.W., Sellen, A., et al. One Is Not Enough: Multiple Views in a Media Space. *Proc. CHI 1993*, ACM Press, 335–341.

[14] Greenhalgh, C. Equip: A Software Platform for Distributed Interactive Systems. *Equator Technical Report 02-002*, University of Nottingham, 2002

[15] Heath, C. and Luff, P. *Technology in Action*. Cambridge University Press, Cambridge, 2000.

[16] Jefferson, G. Transcript Notation. In: Atkinson, J.M. and Heritage, J., eds. *Structures of Social Action: Studies in Conversation Analysis*, Cambridge University Press, 1984, ix-xvi.

[17] Kuzuoka, H., Kosuge, T., et al. Gesturecam: A Video Communication System for Sympathetic Remote Collaboration. *Proc. CSCW 1994*, ACM Press, 35–43.

[18] Kuzuoka, H., Oyama, S., et al. Gestureman: A Mobile Robot That Embodies a Remote Instructor's Actions. *Proc. CSCW 2000*, ACM Press, 155–162.

[19] O' Conner, M., Cosley, D., et al. Polylens: A Recommender System for Groups of Users. *Proc. ECSCW 2001*, Kluwer, 199–210.

[20] Paulos, E. and Canny, J. Ubiquitous Tele-Embodiment: Applications and Implications. *Int. J. Human-Computer Studies*, 46 (6). 861--877.

[21] Rhodes, B. Using Physical Context for Just-in-Time Information Retrieval. *IEEE Trans. Computers*, 52 (8), 1012–1014, 2003.

[22] Sacks, H. *Lectures on Conversation: Vol 1 & 2*. Basil Blackwell, Oxford, 1995.

[23] Simmel, G. and Hughes, E.C. The Sociology of Sociability. *American Journal of Sociology*, 55(3), 254-261.

[24] Urry, J. *The Tourist Gaze: Leisure and Travel in Contemporary Societies*. Sage, New York, 1990.

[25] Woodruff, A., Aoki, P.M., et al. Electronic Guidebooks and Visitor Attention. *Proc. Int'l Cultural Heritage Informatics Meeting*, Milan, 2001, 437-454.

Sharing Context History in Mobile, Context-Aware Trails-Based Applications

Mike Spence, Cormac Driver, Siobhán Clarke
Department of Computer Science
Trinity College Dublin
Dublin 2, Ireland
{spencem, driverc, sclarke}@cs.tcd.ie

ABSTRACT
The growth of ubiquitous computing has given rise to a range of possibilities for context-based application development. The Hermes project is addressing the development of a generic framework to support the design and implementation of mobile, context-aware applications by focusing on the core abstraction of a trail. This paper discusses a major element of our current work - augmenting the Hermes framework with collaborative context and context history components. These components will give application developers the ability to obtain, manage, and exploit current and historical context information, such as trail histories, that can only be acquired via collaboration between a number of dedicated sensors and devices in the ubiquitous computing space.

Keywords
Trails, collaborative context, context history, mobile devices, context-aware, ubiquitous computing, framework development

INTRODUCTION
The development of ubiquitous computing applications poses numerous challenges to software developers. Many issues inherent to the ubiquitous computing paradigm must be tackled during each application development effort, meaning that developers repeatedly encounter the same or similar issues, regardless of the application under consideration. These issues range from low-level programming issues to high-level usability issues. Notable examples of such issues include intermittent network connectivity and executing resource-intensive application adaptation algo-rithms on resource-constrained devices.

Hermes (http://hermes.dsg.cs.tcd.ie) is a software framework for mobile, context-aware trails-based applications that will support developers by providing generic components containing structure and behaviour common to all trails-based applications [2]. Mobile, context-aware applications are those that run on mobile devices, such as PDAs or smartphones, and have an awareness of the physical and social situation in which they are deployed. *Context* is defined as any information that can be used to characterize this situation [3]. A *trail* is a collection of selected activities, together with associated locations and other information, and a dynamically reconfigurable recommended visiting order.

Trails underpin a wide range of useful applications for a mobile user who has a set of activities that may or should be carried out throughout the day at different locations. Combining the trails concept with mobile, context-aware technology creates opportunities for innovative activity-based application development. Examples of trails-based applications that are both mobile and context-aware include tour guides, courier support/management systems, basic route planners, treasure hunt games and student activity support systems.

This paper discusses *collaborative context* - the sharing of context data between devices in a ubiquitous computing environment and *trail histories* – a recorded account of how a user makes use of a trail. We have recently begun work on adding collaborative context and context history comp-onents to the existing Hermes framework. The remainder of this paper contains an overview of the collaborative context concept and its challenges; a discussion of trails, trail histories, and their challenges; and concludes with a presentation of planned future work and open questions.

COLLABORATIVE CONTEXT
Fundamental to ubiquitous computing applications is a requirement to sense or obtain relevant information from the user's environment. In a mobile, context-aware trails-based situation, that environment can be spread over a relatively large geographical space e.g., a city, and the types of context information required include those not directly provided by typical dedicated sensors e.g., crowd density at distant locations on a user's trail and ratings of activities by users. For this paper, we differentiate *dedicated sensors*, such as accelerometers, thermometers, or GPS sensors, from the higher-level *devices*. Dedicated sensors are built to sense a single aspect of the environment and communicate that value. A device might have a sensor

embedded in it, but the property of being able to combine that sensor information with other information makes it a device. Devices include servers, laptops, mobile devices, and other computing platforms that are not dedicated sensors.

Collaborative context is the subset of context that is acquired via the communication between dedicated sensors and devices in a ubiquitous computing space and can be used to increase both the range of context available and context reliability. More formally, collaborative context is defined recursively as:

- Context information acquired directly from other devices that are not dedicated sensors
- Context information acquired from dedicated sensors through other devices
- Context information derived from the combination of *collaborative context* and possibly sensor data and/or context information already on the device

Context obtained directly from dedicated sensors is not included in collaborative context, because another device does not have access to this context; this means another device *cannot act on the context data before it is received*. This property of collaborative context is the important element in the challenges presented in the next section

An example of using collaborative context in an application is sharing trail histories to revise activity or trail details. A trail history is a recorded account of how the user makes use of a given trail. This includes the initial application-generated trail, any reconfigurations of that trail, the trail actually followed by the user, and other context information that might aid in explaining why the user might have deviated from or followed the trail. If an application on a user's device receives this historical context information from other devices, it can infer from it, for example, that a particular activity took longer than originally expected. The application might then lengthen the predicted duration of that activity. This form of collaborative technique can be used to provide a more realistic trails experience to mobile users.

Collaborative Context Challenges

The use of collaborative context has the potential to greatly improve both the manner in which trails are generated and the representation of the ubiquitous computing space to the user; however, there are a number of significant research challenges in this area. Rather than concentrating on the communication challenges for requesting and transferring context between devices, our research in collaborative context focuses on how context information is disclosed and how context information is refined upon acquiring it. Consequently, these are the three challenges of collaborative context that we are addressing:

Privacy

Privacy addresses the conditions under which an application stores or discloses a piece of information. Some applications will not function without being provided some form of context information, even if it is less precise e.g., disclosing city or country instead of exact GPS coordinates. As discussed in [5], most users will only agree to provide a system with personal data if the benefits gained from the disclosure outweigh the risks e.g., purchasing items by credit card on the Internet.

Trust

Trust is the amount of confidence a party has that another party will behave as expected. Trust in relation to collaborative context is a function of the confidence in the original source of the context information, as well as any other device it passes through en route to the destination. It can also vary based on the type of the context information.

Perception

The term perception is used to describe the difficulties of relating a piece of context from an outside environment into the device's environment. In collaborative context, the originators or intermediary manipulators of context information are normally in a different situation than the receivers of that context. Whether this difference is that of location, application goals, or a combination of factors, the context a device communicates depends on its perception of the surrounding environment. Unfortunately, defining and communicating the entire relevant context is impossible for most applications. Imprecision, staleness, conflicting, incor-rect, or missing values, and subjective opinions and recommendations are all important perception issues.

To some extent, these are important challenges of acquiring and using any type of context; however, due to the indirect path and possibly subjective nature of the pieces of context, handling these issues is essential for collaborative context.

TRAILS AND TRAIL HISTORIES

Current context is not the only source of useful collaborative context to an application; historical context can also be an important source of shared context. A key component of collaborative context in trails-based applications is the ability to share past and current trails. Some benefits gained by the ability to use and share trails and trail histories are:

Avoiding trail generation

Due to the limited processing power of the mobile devices and the processing-intensive nature of trail generation, using a previously recorded trail to avoid calculating the entire trail again can have beneficial effects on trail quality and application responsiveness.

Verifying assumptions

Trail histories can be used as an aid in verifying or revising certain assumptions made by the application about the activities and the trail as a whole, such as activity duration or journey time between activities.

Utilizing feedback for adaptation

Trail generation utilizes user preference and other context information to predict in what order and which route users will want to experience a trail. Trails are unique to most predictive context-aware applications in that there is implicit feedback on whether the user is satisfied with the current generated trail i.e., if the user follows the trail. This feedback can aid in predicting under what conditions a user might deviate from a particular trail, and an application can use that information to generate a more acceptable trail.

Synchronizing users

Trails and trail histories may be shared among friends and coworkers to allow users to synchronize their activities and routes between activities. Current trails can be used to determine where a user plans to be, and trail histories can be used to predict where a user might be.

Offloading trail computation

Another device might have greater knowledge of the activities or routes between the activities. In order to leverage the knowledge of that device, part of a trail or a collection of activities can be given to that device to order the activities more appropriately or choose more suitable routes. This is also an indirect method of using information contained in another device's trail histories without the overhead of actually passing the trail histories themselves. Additionally, if groups of activities can be partitioned for ordering like this, the runtime complexity of the trail generation can be significantly reduced; however, finding partitions generically is a nontrivial task [4].

Trails and Trail Histories Challenges

Sharing historical context creates challenges, because the information is not current and the interaction occurred in another device with a different user context. Consequently, even though sharing this information has important privacy and trust issues, representing and utilizing trail histories deal mostly with the perception challenge.

Challenges in representing trail histories

The choice of what context information to store in the representation of a trail history is vital for trail reuse, as this additional context might help explain why an application generated a particular trail or why a user might have deviated from it. If the application that created the trail history includes a useful subset of motivating context information, this removes the trail history exploiter, which might also be the trail history creator, from attempting to reconstruct that information itself. However, discovering an optimal subset of available context information to store in a trail history can be difficult as the most useful subset can be specific to the exploiting application's user, device, and situation.

Trail history size is also an important representation concern when considering the limited storage space and low-bandwidth communications on mobile devices. Consequently, finding the optimal subset of the available context information to store in a trail history is also crucial for use by resource-constrained devices. To alleviate the size concerns, an application could store or pass only a subset of the trail history; however, determining the correct subset for later, possibly off-device, reuse can be difficult, and this still involves the issues associated with finding a subset of additional context to include.

Dynamic trail reconfiguration presents a final trail history representation challenge. As a user follows a trail, minor adjustments such as an activity running a few minutes over or a route between activities not taking as long to traverse as expected might be useful to store. More dramatic adjustments such as an added activity or a route closure might force the activity ordering to change. Finding a way to represent these changes in way that is efficient for applications to examine and is sensitive to the resource-constrained nature of mobile devices is essential.

Challenges in utilizing trails and trail histories

Other challenges to actually utilizing generic trails and trail histories, rather than just representing them, are:

- Analyzing trails: Determining what portions of the trail are relevant can be difficult, particularly when they contain activities unfamiliar to an application.
- Comparing trails: Different trails probably have different orderings based on the users' preferences and environment conditions at the time of creation and use. They might also contain different activities.
- Merging trails: This is closely related to analyzing and comparing trails, which are necessary components for merging. While merging multiple trails into one might prevent an application from recalculating an entire trail, it still must find a way to mix and connect the activities in the various trails. There is also a danger that merging might be more expensive than simply generating the trail.

Like most user interactions with their environment in ubiquitous computing, the underlying issue in most of these challenges is that trails and trail histories are complicated pieces of derived context and are influenced by a combination of several factors that are difficult to capture and reuse.

CURRENT WORK AND OPEN ISSUES

A trails-based multi-player riddle game based in Dublin city is currently under development. The object of the game is to collect more points than the other players by the end of the game. A player receives points by answering riddles at a collection of locations and a correct answer removes the riddle at that location. There are a limited number of riddles at each location, all with varying types and point values. The attempts to answer riddles are the activities in the trail and the trail generation and reconfiguration are based on current game state and player-defined strategy preferences.

This application will be used as a platform upon which to design and implement collaborative context and context history components that will aid the enhancement of

existing Hermes framework components. The collaborative context component will be used to pass context information such as game state and trail histories.

Most of the applications of trail histories to the riddle game will be to aid the generation of trails and improve game strategies. Part of our evaluation will be for users to play the game multiple times. Both the user's and other successful players' trail histories from previous games can be analyzed and exploited to create more successful, customized player trails.

Game simplifications, like a fixed collection of previously defined riddle activities, limited geographic space, and a fixed time limit should allow us to focus on a subset of the difficult challenges presented earlier, such as game-specific trail history representation and exploitation on constrained-resource devices.

Allowing a user to play multiple times will also help us answer a key question concerning mobile, trails-based applications: While studies such as the GUIDE project [1] have shown that a one-time use of a trails-based application in an unfamiliar environment can be a great aid to users, once the users are familiar with the geographic area and the activities, will the trails-based application be useful? Our initial feeling is that this will depend on the quality of the context information acquired and how an application utilizes it to aid users in their tasks. If players continue to use the trails-based application features even after becoming familiar with the area and activities, this will certainly be a positive step in demonstrating the utility of trails-based applications.

ACKNOWLEDGMENTS

Mike Spence is funded by Intel Corporation. Cormac Driver is supported by the IRCSET Embark Initiative.

REFERENCES

1. Cheverst, K. et al. "Experiences of Developing and Deploying a Context-Aware Tourist Guide: The GUIDE Project", in *Proceedings of sixth ACM International Conference Mobile Computing and Networking (MOBICOM'2000)*, Boston, U.S., ACM Press, pp. 20-31, August 2000.
2. Clarke, S. and Driver, C. "Context-Aware Trails". *IEEE Computer*, Vol. 37, No. 8. pp. 97-99, August 2004.
3. Dey A. and Abowd G., "Towards a Better Understanding of Context and Context-Awareness", *Proceedings of the CHI 2000 Workshop on The What, Who, Where, When, and How of Context-Awareness,* The Hague, Netherlands, April 1-6, 2000.
4. Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2002.
5. Schilit B. et al. "Ubiquitous location-aware computing and the 'Place Lab' Initiative". *In Proceedings of the First ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspot*. San Diego, CA, Sept. 2003.

Author Biographies

Mike Spence received his B.A. in Computer Science from Rice University in 2000. After working for Scient Corporation and Closest Point Solutions as a developer, he returned to school and received a Postgraduate Diploma in Ubiquitous Computing from Trinity College Dublin. He is currently a PhD student in the Distributed Systems Group at Trinity College Dublin under his supervisor, Siobhán Clarke.

Cormac Driver holds a B.Sc. in Business Information Systems from the Dublin Institute of Technology and a M.Sc. in Networks and Distributed Systems from Trinity College Dublin. He is currently a PhD student in the Distributed Systems Group at Trinity under Siobhán Clarke and is working on the Hermes project.

Siobhán Clarke is a lecturer in the Department of Computer Science at Trinity College Dublin. Her research interests are in design and programming models for mobile, context-aware computing.

Next Location Prediction Within a Smart Office Building

Jan Petzold, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer

University of Augsburg
Institute of Computer Science
Eichleitnerstr. 30, 86159 Augsburg, Germany
{Petzold, Bagci, Trumler, Ungerer}@Informatik.Uni-Augsburg.DE

ABSTRACT

We investigate the feasibility of in-door next location prediction using sequences of previously visited locations and compare the efficiency of several prediction methods. The scenario concerns employees in an office building visiting offices in a regular fashion over some period of time. We model the scenario by different prediction techniques like Neural networks, Bayesian networks, State and Markov predictors. We use exactly the same evaluation set-up and benchmarks to compare the different methods. The publicly available Augsburg Indoor Location Tracking Benchmarks are applied as predictor loads.

Keywords

context awareness, location prediction, proactive

1. INTRODUCTION

We investigate to which extend the movement of people working in an office building can be predicted based on room sequences of previous movements. Our hypothesis is that people follow some habits, but interrupt their habits irregularly, and sometimes change their habits. Moreover, moving to another office fundamentally changes habits too.

Our aim is to investigate how far machine learning techniques can dynamically predict room sequences, time of room entry, and duration of stays independent of additional knowledge. Of course the information could be combined with contextual knowledge as e.g. the office time table or personal schedule of a person, however, at this time we focus on dynamic techniques without contextual knowledge.

Further interesting questions concern the efficiency of training of a predictor, before the first useful predictions can be performed, and of retraining, i.e. how long it takes until the predictor adapts to a habitual change and provides again useful predictions. Predictions are called useful if a prediction is accurate with a certain confidence level (see [18] for confidence estimation of state predictors).

Moreover, memory and performance requirements of a predictor are of interest in particular for mobile appliances with limited performance ability and power supply.

The predictions could be used for a number of applications in a smart office environment. We demonstrate two application scenarios:

- In the Smart Doorplate Project [22] a visitor is notified about the probable next location of an absent office owner within a smart office building. The prediction is needed to decide if the visitor should follow the searched person to his current location, go to the predicted next location, or just wait till the office owner comes back.

- A phone call forwarding to the current office location of a person is an often proposed smart office application, but where to forward a phone call in case that a person just left his office and did not yet reach his destination? The phone call could be forwarded to the predicted room and answered as soon as the person reaches his destination.

Our experiments as part of Smart Doorplate Project yielded a collection of movement data of four persons over several months that are publicly available as Augsburg Indoor Location Tracking Benchmarks [13, 14]. We use this benchmark data to evaluate several prediction techniques and compare the efficiency of these techniques with exactly the same evaluation set-up and data. Moreover, we can estimate how good next location prediction works - at least for the Augsburg Indoor Location Tracking Benchmark data.

2. RELATED WORK

The Adaptive House project [11] of the University of Colorado developed a smart house that observes the lifestyle and desires of the inhabitants and learned to anticipate and accommodate their needs. Occupants are tracked by motion detectors and a neural network approach is used to predict the next room the person will enter and the activities he will be engaged. Hidden Markov models and Bayesian inferences are applied by Katsiri [8] to predict people's movement. Patterson et al. [12] presented a method of learning a Bayesian model of a traveller moving through an urban environment based on the current mode of transportation. The learned model was used to predict the outdoor location of the person into the future.

Markov Chains are used by Kaowthumrong et al. [7] for active device selection. Ashbrook and Starner [1] used location context for the creation of a predictive model of user's future movements based on Markov models. They propose to deploy the model in a variety of applications in both single-user and multi-user scenarios. Their prediction of future location

is currently time independent, only the next location is predicted. Bhattacharya and Das [3] investigate the mobility problem in a cellular environment. They deploy a Markov model to predict future cells of a user.

An architecture for context prediction was proposed by Mayrhofer [10] combining context recognition and prediction. Active LeZi [4] was proposed as good candidate for context prediction.

All approaches perform location prediction with specific techniques and scenarios. None covers a smart office scenario and none compares several prediction techniques. Moreover, none of the evaluation data is publicly available. Therefore the applied techniques are hard to compare.

3. AUGSBURG INDOOR LOCATION TRACKING BENCHMARKS

The Augsburg Indoor Location Tracking Benchmarks were derived within the Smart Doorplate project [22] which acts as testbed for implementation and evaluation of the proposed prediction techniques. A Smart Doorplate shows information about the office owner like a traditional static doorplate. The Smart Doorplate, however, additionally shows dynamic information like the presence or absence of the office owners. If an office owner is absent from his office the doorplate directs a visitor to the current location of the absent office owner. Furthermore it predicts the next location of the absent office owner and the entering time of this location. This additional information can help the visitor to decide whether he follows the office owner or waits for him.

The predicted location information can also be used for switching over the phone to the next location of a clerk. That means when the clerk leaves his office, the system predicts the next location of the clerk and switches over the phone call to this location. As example we consider a scenario with Mr. A. and Mr. B.:

> Mr. A. leaves his office and the system predicts the office of Mr. B. as next location. Now Mr. A. is en route to this office.
>
> In Mr. B.'s office the phone rings. He answers the call and says: "No, Mr. A. isn't here." At this moment Mr. A. enters the office of Mr. B. and Mr. B. speaks to the caller: "Oh however, Mr. A. is now here. I give over the phone."

To evaluate prediction techniques in the two described scenarios we needed movement sequences of various clerks in an office building. Therefore we recorded the movements of four test persons within our institute building and packaged the data in the *Augsburg Indoor Location Tracking Benchmarks* [13, 14].

We collected the data in two steps, first we performed measurements during the summer term and second during the fall term 2003. In the summer we recorded the movements of four test persons through our institute over two weeks. The summer data range from 101 to 448 location changes. Because this data was too short we started a further measurement with the same four test persons in the fall. Here we accumulated date over five weeks. The fall data range from 432 to 982 location changes. These benchmarks will be used for evaluating the different prediction techniques in the described scenarios.

4. COMPARISON OF PREDICTION TECHNIQUES

Several prediction techniques are proposed in literature – namely Bayesian networks [6], Markov models [2] or Hidden Markov models [21], various Neural network approaches [5], and the State predictor methods [19]. The challenge is to transfer these algorithms to work with location sequences.

We currently investigate Neural networks, Bayesian networks, Markov and State predictors. First we chose from the multitude of Neural networks the most well-known, the multi-layer perceptron with one hidden layer and back-propagation learning algorithm. The multi-layer perceptron was chosen because of its general application domain and its popularity in the Neural network research community. Details on the multi-layer perceptron with back-propagation learning were published in [23]. After analyzing more neural networks we decided that an Elman net fits better for solving the next location problem. Elman nets hold a so-called context layer. With this layer the nets are suited to learn sequences. Recent results show that Elman nets are usually better suited than the multi-layer perceptron [9].

In the case of Bayesian networks we started with a static Bayesian network. Afterwards, in order to predict a future context of a person, the usage of a dynamic Bayesian network was chosen. This network consists of different time slices which all contain an identical Bayesian network. Bayesian networks are particularly well suited to model time [20].

The state predictor method originates in branch prediction and data compression algorithms that are transformed and adapted to fit the scenario of context prediction. Generally speaking, the prediction principle is derived from Markov chains theory [2]. In [15, 16, 17] several one- and two-level predictors were proposed and evaluated by synthetic benchmarks. In [19] the state predictors were evaluated with the Augsburg Indoor Location Tracking Benchmarks. Moreover we evaluated the well-known Markov predictor.

Table 1 compares the prediction accuracies of the Neural networks Elman net and multi-layer perceptron (MLP), Bayesian network, State predictor, and Markov predictor showing always the best results yielded for each person. The configurations may vary for different person. The configuration details are published in the papers cited above. Typically, there is no superb configuration of a predictor for all persons. The shown prediction accuracies are derived for the first scenario where a visitor will be informed about the potential return of an office owner. That means the accuracies include only predictions when the employee isn't in his own room. Furthermore the following set-up was used: All prediction algorithms were trained with summer data and the accuracies were measured with the fall data (see section 3). The results show that there isn't a universal predictor.

Because of the sometimes unreliable results of predictions it may be sometimes better to make no prediction instead

Table 1: Prediction accuracies of the up to now evaluated prediction techniques

	Elman net	MLP	Bayesian network	State predictor	Markov predictor
Person A	91.07%	87.39%	85.58%	88.39%	90.18%
Person B	78.88%	75.66%	86.54%	80.35%	78.97%
Person C	69.92%	68.68%	86.77%	75.17%	75.17%
Person D	78.83%	74.06%	69.78%	76.42%	78.05%

of a wrong prediction. Humans may be frustrated by too many wrong predictions and won't believe in further predictions even when the prediction accuracy improves over time. Therefore confidence estimation of context prediction methods is necessary. In [18] three confidence estimation techniques for the state predictor method were proposed and evaluated. The proposed confidence estimation techniques can also be transferred to other prediction methods like Markov Predictors, Neural network, or Bayesian networks.

Moreover, also the length of stay is of interest. This can easily be predicted by dynamic Bayesian networks or attached to other predictors as arithmetic mean or median of previous length of stay in the respective room.

5. CONCLUSION AND FUTURE WORK

We evaluated several prediction techniques for indoor location prediction with exactly the same set-up and data. The evaluation shows a variation of prediction accuracies among the different prediction methods as well as within configurations of a specific methods. Prediction accuracies of 70% to 90% could be reached.

In future we will analyze more prediction techniques which could solve the problem of next location prediction, e.g. Hidden Markov models. Furthermore we will develop different hybrid predictor. A hybrid predictor holds a set of simple predictors and chooses a predictor to predict the next location on the basis of a selection criteria. Moreover, we will include length of stay and daytime in all predictors. Also we will generate more benchmark data by an automatic location tracking system.

The prediction algorithms should also be evaluated with other context domains. For example outdoor movement patterns can be used to predict the next region a person will enter. Elevator prediction could anticipate at which floor an elevator will be needed next. Routing prediction for cellular phone systems may predict the next radio cell a cellular phone owner will enter based on his previous movement behavior. The main problem is to get appropriate benchmark data.

REFERENCES

[1] Daniel Ashbrook and Thad Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.

[2] Ehrhard Behrends. *Introduction to Marcov Chains*. Vieweg, 1999.

[3] Amiya Bhattacharya and Sajal K. Das. LeZi-Update: An Information-Theoretic Framework for Personal Mobility Tracking in PCS Networks. *Wireless Networks*, 8:121–135, 2002.

[4] Karthik Gopalratnam and Diane J. Cook. Active LeZi: An Incremental Parsing Algorithm for Sequential Prediction. In *Sixteenth International Florida Artificial Intelligence Research Society Conference*, pages 38–42, St. Augustine, Florida, USA, May 2003.

[5] Kevin Gurney. *An Introduction to Neural Networks*. Routledge, 2002.

[6] Finn V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, 1996.

[7] Khomkrit Kaowthumrong, John Lebsack, and Richard Han. Automated Selection of the Active Device in Interactive Multi-Device Smart Spaces. In *Workshop at UbiComp'02: Supporting Spontaneous Interaction in Ubiquitous Computing Settings*, Gteborg, Sweden, 2002.

[8] Eleftheria Katsiri. Principles of Context Inference. In *Adjunct Proceedings UbiComp'02*, pages 33–34, Gteborg, Sweden, 2002.

[9] Mirjam Kuhlmann. Untersuchung von Neuronalen Netzen zur Kontextvorhersage in ubiquitären Systemen. Master's thesis, Institute of Computer Science, University of Augsburg, Germany, February 2005. In German.

[10] Rene Mayrhofer. An Architecture for Context Prediction. In *Advances in Pervasive Computing*, number 3-85403-176-9. Austrian Computer Society (OCG), April 2004.

[11] Michael C. Mozer. The Neural Network House: An Environment that Adapts to its Inhabitants. In *AAAI Spring Symposium on Intelligent Environments*, pages 110–114, Menlo Park, CA, USA, 1998.

[12] Donald J. Patterson, Lin Liao, Dieter Fox, and Henry Kautz. Inferring High-Level Behavior from Low-Level Sensors. In *5th International Conference on Ubiquitous Computing*, pages 73–89, Seattle, WA, USA, 2003.

[13] Jan Petzold. Augsburg Indoor Location Tracking Benchmarks. Technical Report 2004-9, Institute of Computer Science, University of Augsburg, Germany, February 2004. http://www.informatik.uni-augsburg.de/skripts/techreports/.

[14] Jan Petzold. Augsburg Indoor Location Tracking Benchmarks. Context Database, Institute of Pervasive Computing, University of Linz, Austria. http://www.soft.uni-linz.ac.at/Research/Context_Database/index.php, January 2005.

[15] Jan Petzold, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. Context Prediction Based on Branch Prediction Methods. Technical Report 2003-14, Institute of Computer Science, University of Augsburg, Germany, July 2003. http://www.informatik.uni-augsburg.de/skripts/techreports/.

[16] Jan Petzold, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. Global and Local Context Prediction. In *Artificial Intelligence in Mobile Systems 2003 (AIMS 2003)*, Seattle, WA, USA, October 2003.

[17] Jan Petzold, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. The State Predictor Method for Context Prediction. In *Adjunct Proceedings Fifth International Conference on Ubiquitous Computing*, Seattle, WA, USA, October 2003.

[18] Jan Petzold, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. Confidence Estimation of the State Predictor Method. In *2nd European Symposium on Ambient Intelligence*, pages 375–386, Eindhoven, The Netherlands, November 2004.

[19] Jan Petzold, Faruk Bagci, Wolfgang Trumler, Theo Ungerer, and Lucian Vintan. Global State Context Prediction Techniques Applied to a Smart Office Building. In *The Communication Networks and Distributed Systems Modeling and Simulation Conference*, San Diego, CA, USA, January 2004.

[20] Jan Petzold, Andreas Pietzowski, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. Prediction of Indoor Movements Using Bayesian Networks. In *Location- and Context-Awareness (LoCA 2005)*, Oberpfaffenhofen, Germany, May 2005.

[21] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *IEEE*, 77(2), February 1989.

[22] Wolfgang Trumler, Faruk Bagci, Jan Petzold, and Theo Ungerer. Smart Doorplate. In *First International Conference on Appliance Design (1AD)*, Bristol, GB, May 2003. Reprinted in Pers Ubiquit Comput (2003) 7: 221-226.

[23] Lucian Vintan, Arpad Gellert, Jan Petzold, and Theo Ungerer. Person Movement Prediction Using Neural Networks. In *First Workshop on Modeling and Retrieval of Context*, Ulm, Germany, September 2004.

AUTHOR BIOGRAPHIES

Jan Petzold received his diploma in Business Mathematics from the University of Augsburg in 2001. Since 2002 he works as research assistant in the Systems and Networking Group at the University of Augsburg.

Faruk Bagci received his diploma in Computer Science from the University of Aachen in 1999. From 2000 to 2001 he worked as research assistant at the University of Aachen. In 2001 he changed to the Systems and Networking Group at the University of Augsburg.

Wolfgang Trumler received his diploma in Computer Science in 1996 at the University of Karlsruhe. He was employed for two years as a developer for integrated circuits before he founded the company Intermediate GmbH & Co. KG in Karlsruhe. In 2002, after four years as CTO at Intermediate, he started as a research assistant at the Chair for Systems and Networking at the University of Augsburg.

Theo Ungerer is Chair of Systems and Networking at the University of Augsburg, Germany. Previously he was Professor for Computer Architecture at the Technical University of Karlsruhe 1993-2001. He received a doctoral degree in 1986 and a second doctoral degree (Habilitation) in 1992 at the University of Augsburg. His research interests are in the areas of embedded and ubiquitous computing.

Tracking Personal Histories for Knowledge Discovery Tasks

Dennis P. Groth
School of Informatics
Bloomington, Indiana 47408-3912
dgroth@indiana.edu

ABSTRACT

Interactive visualizations provide an ideal setting for exploring the use and exploitation of personal histories. Even though visualizations leverage innate human capabilities for recognizing interesting aspects of data, it is unlikely that two users will follow the exact process for discovery. This results in an inability to effectively recreate the exact conditions of the discovery process, which we call the knowledge rediscovery problem. Because we cannot expect a user to fully document each of their interactions, there is a need for visualization systems to maintain user trace data in a way that enhances a user's ability to communicate what they found to be interesting, as well as how they found it. This project presents a model for representing user interactions that articulates with a corresponding set of annotations, or observations that are made during the exploration. This problem is only made more challenging when pervasive computing and corresponding interactions across devices is factored in.

Keyword

Provenance, history tracking, annotation, collaboration.

INTRODUCTION

Visualization research is frequently presented in terms of a graphic image of a visual representation, along with a verbal description of what the observer should recognize. While this traditional approach in reporting results is necessary and meaningful, it is important to note that, in presenting the result in such a way, the researcher is not reporting a visualization. Rather, the researcher is reporting a presentation graphic. While it is necessary to generate a presentation graphic to report the results, the graphic is substantially inadequate for other researchers or practitioners to apply the results due to the static nature of the information presented in this form.

The final presentation of the visualization output results from a stream of actions performed against the data. A common view of the transformation process is represented by the visualization pipeline, as shown in Figure 1. Under this model, data may be transformed through any number of processes prior to display. The output then may undergo an indeterminate number of user specified view transformations. If we assume a 3D representation, the most basic view transformations are rotation, translation and scaling, with others supported by specific applications.[3]

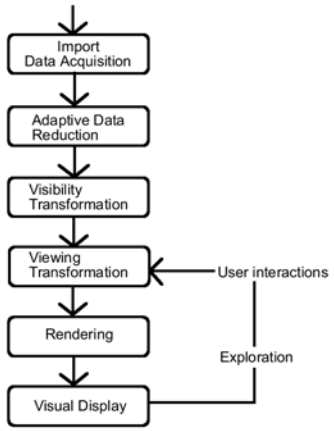


Figure 1: Visualization pipeline.[15]

The main contribution of this research is a conceptual model of user interaction and observation for data visualization. The model is *generic*, in the sense that it concisely captures changes to the state of the visualization made by the user in a way that provides recall of steps the user took to achieve the visual representation. In addition to the conceptual model we describe an instantiation of the model, demonstrating how the model can be adapted to support a variety of modalities of interaction tracking. A prototype implementation of the model is used to demonstrate how the model can be used to enhance user navigation.

Our intent is to consider the interactions with data as knowledge itself. Armed with this knowledge, researchers will be in a position to share not only what was found to be interesting (the discovery), but exactly how it was found (the discovery process). In a broader context, we consider

this research to be a fundamental contribution to developing solutions for an emergent research area: *information provenance*.[8] Problems in knowledge and data provenance[2] are gaining interest, with broad applications to the advancement of scientific discovery [13].

Provenance is a term that refers to the lineage of an item. While some people associate the term with artwork, and the lineage of who owned, or possessed the piece, we use it in the context of the information discovery process. The model that we are presenting supports provenance by fully documenting the discovery process. The prototype demonstrates how users can interact with the history of interactions and capture annotations in the same context. Another user may take the interaction data and use it against a different dataset, to see how general the technique may be.

RELATED WORK

The interaction model builds upon work from three areas of research: knowledge discovery in databases (KDD), annotation; and user tracking. We make an assumption that the knowledge discovery process is, indeed, a process, and that the steps that a user takes to discover knowledge are as important as the knowledge itself. Described by Fayyad, et al [5], the KDD process is frequently depicted in terms of a number of iterative steps. There are, of course, obvious similarities between the KDD process and the data visualization pipeline.[15] Our approach is to track all interactions with the data during the KDD process and to provide visualization tools for the KDD process.

A critical aspect of the process is the implied interaction with a user. Obviously, the user is involved with problem selection, as well as the interpretation of the results. Often, the user may review the results and develop a more refined problem statement, which initiates further exploration. In the context of this research, annotation is the adding of information to existing data by a user. For visualizations, numerous approaches have been described. Marshall and Brush [12] discuss the issue of shared annotations.

User tracking involves the recording of actions taken by users in the course of completing a task. Scaife and Rogers [14] critically examine the linkages between external representations (e.g. visualizations) and a user's corresponding internal representation of the information. They describe the concept of a cognitive trace, which may include explicit marks, or highlighting, of information. A need to record the corresponding parameter settings for the software is also identified. Fitzgerald, et al [6] define a framework for describing event-tracking for multimodal user interfaces. Such a framework can be used to develop a more comprehensive model of user interaction. Franklin, et al [7] describe a tracking mechanism for an electronic classroom environment, in which the users actions are tracked for playback purposes.

Tracking of user interactions within a visualization environment has been studied by Lee and Grinstein [10] and, more recently, Jankun-Kelly, et al [9] and Lowe, et al [11]. These previous efforts are particularly relevant to our work at the conceptual level. Where we differ from them is in the capturing of meta-information, such as annotations, along with the interactions.

CONCEPTUAL MODEL AND PROTOTYPE

We base the model [8] on directed graphs, with nodes signifying measurable states of the system, and edges denoting transitions between the states. The states of the system are generically captured in the model, leaving it up to the implementation to define the specific contents of the state and transition information. For example, as described in [9], the transitions might contain discrete interactions, such as zoom, rotate, or translate. Pictorially, the graph can be depicted as shown in the example in Figure 2.

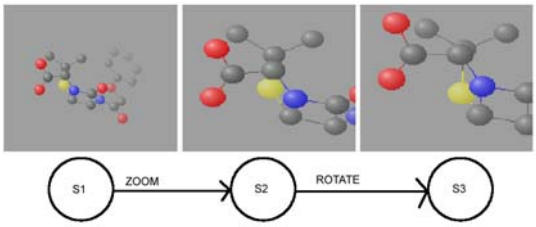


Figure 2: A simple interaction graph and user views.

The model allows for the articulation of annotation data with the interactions that are being applied to the visualization. For example, we will assume that the user follows the general process of: 1) observing the display; 2) making an annotation; and 3) applying an interaction. For this example, the annotation data is represented by text. However, there is no restriction on the mode of input used to perform the annotation.

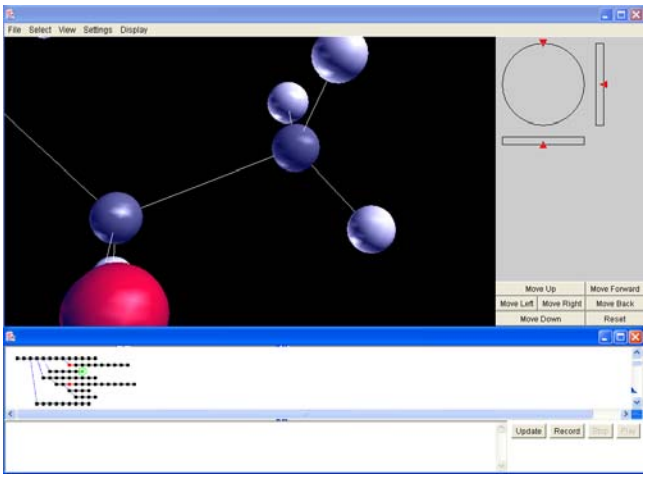


Figure 3: Prototype implementation and the history panel.

Our prototype is implemented according to the architecture shown in Figure 4. We modified a system for visualizing multivariate data to have the CheckState, GetState and SetState methods to generate the graph nodes. The state information that we tracked within the system was a viewpoint model - camera position and direction within a 3D environment.

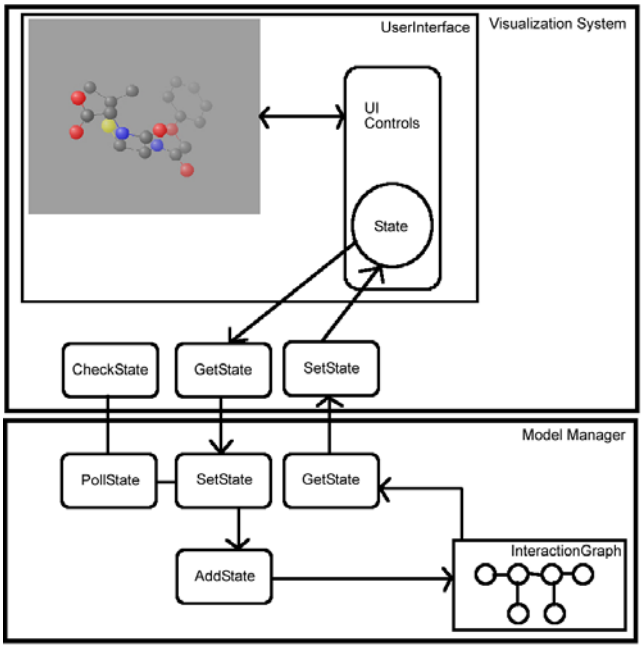


Figure 4: Architecture for tracking interaction history.

One of the most interesting aspects of our prototype is the model manager interface, which exposes the interaction graph to the user. The resulting application allows the user to interact with the visualization system as well as the interaction graph. Figure 3 shows a screen capture from the model manager. The user interaction was a simple sequence of zooming operations to display an overview of the entire dataset.

The prototype model manager supports annotation directly through either typed comments, or recorded voice. This capability saves the visualization system the effort of performing annotation capabilities. We have developed a tablet PC based interface to support direct scribbling of annotations.[4] The interaction graph displays visual cues to indicate current position within the interaction history as well as the location of annotations.

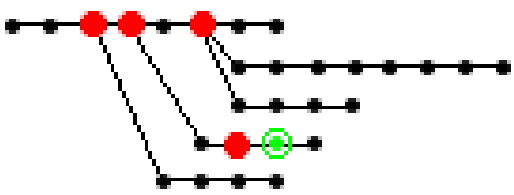


Figure 5: An example interaction graph.

It is worth pointing out that there is no restriction in the model for branching, or non-linear behavior represented in the interaction graph. However, in order for the model to support non-linear behavior tracking the model manager must keep track of where the user is relative to the interaction graph. The prototype supports this feature by creating branches in the graph if the current state is not a leaf node. An example graph is shown in Figure 5. Note the use of color to provide visual cues to the user. Larger, red nodes signify the location of annotation data, a single green node with a larger circle around the node is the current location in the graph.

Users can edit their histories by cutting segments of the graph out with the mouse. Histories can be combined by inserting graphs as sub-graphs of an active history. Also, we have developed intelligent pruning techniques to collapse the graph into only annotated nodes.

OPPORTUNITIES AND CHALLENGES

The interaction model demonstrates one way of implementing personal history tracking. There are a number of directions that this project can take. These opportunities are detailed here.

Question: Can interactions be modeled generically?

A generic model of interaction history allows the separation of interactions from the objects that are the target of interaction. The interactions can then be replayed against different datasets, or in different contexts.

Question: To what extent does collaborative, or shared, explorations influence discovery?

Collaborations may occur asynchronously, when users send their histories to each other, or synchronously, when users are simultaneously interacting within a shared space.

Question: Can recommender systems be developed to leverage historical interactions?

The opportunity exists to point users to views that other people have taken of the data. However, we need to determine what makes a particular view “interesting” enough to warrant suggesting.

Question: Should recommender systems be developed to leverage historical interactions?

What personal or privacy concerns are there? Will users only follow the paths of others? If so, exploiting personal histories may actually stifle discovery instead of enabling it. A different view of the problem suggests that coverage of the interaction space can be maximized by suggesting views that have not occurred yet.

Question: Can histories be compared?

One way of comparing histories is to break the interactions into discrete events and disregard temporal relationships. This approach is akin to leaving a fingerprint on an object – we can tell that something has been touched. The temporal aspect, or the context of when objects are

interacted with, is much more interesting. We will need to develop a similarity metric to quantify history comparisons.

CONCLUSION

We have presented a model for tracking of interactions for knowledge discovery tasks. An implementation of the model for visualization tasks shows how personal histories are captured, edited and shared. The prototype further explores how users interactions with their personal histories introduces a new style of interaction. Clearly, there are many interesting and challenging problems to be addressed in this research space.

REFERENCES

1. Bargeron D. and Moscovich, T. Reflowing Digital Ink Annotations. *Proceedings of the Conference on Human Factors in Computing Systems (*2003), 385-393.
2. Buneman, P., Khanna, S., and Tan, W. Why and Where: A Characterization of Data Provenance. *Proceedings of the International Conference on Database Theory* (2001), 316-330.
3. Card, S. and MacKinlay, J. The Structure of the Information Visualization Design Space. *Proceedings of the IEEE Symposium on Information Visualization* (1997), 92-99.
4. Ellis, S. and Groth, D. A Collaborative Annotation system for Data Visualization. *Proceedings of the ACM Conference on Advanced Visual Interfaces.* (2004), 411-414.
5. Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. From Data Mining to Knowledge Discovery: An Overview. *Advances in Knowledge Discovery and Data Mining*. (1996), 1-34.
6. Fitzgerald, W., Firby, R. and Hanneman, M. Multimodal Event Parsing for Intelligent User Interfaces. *Proceedings of the International Conference on Intelligent User Interfaces.* (2003), 53-60.
7. Franklin, D., Budzik, J. and Hammond, K. *Plan-Based Interfaces: Keeping Track of User Tasks and Acting to Cooperate. Proceedings of the International Conference on Intelligent User Interfaces.* (2002), 79-86.
8. Groth, D. Information Provenance and the Knowledge Rediscovery Problem. *Proceedings of the International Conference on Information Visualization.* (2004), 345-351.
9. Jankun-Kelly, T.J., Ma, K., and Gertz, M. A Model for the Visualization Exploration Process. *Proceedings of the IEEE Conference on Visualization.* (2002), 323-330.
10. Lee, J. and Grinstein, G. An Architecture for Retaining and Analyzing Visual Explorations of Databases. *Proceedings of the IEEE Conference on Visualization.* (1995), 101-109.
11. Lowe, N., Tory, M. Potts, S., Datta, A. and Moller, T. A Parallel Coordinates Style Interface for Exploratory Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics.* (2005), 71-80.
12. Marshall, C. and Brush, A. From Personal to Shared Annotations. *Proceedings of the Conference on Human Factors in Computing Systems.* (2002), 812-813.
13. Myers, J., Chappell, A., Elder, M., Geist, A. and Schwidder, J. Reintegrating the Research Record. *IEEE Computing in Science and Engineering.* (2003), 44-50.
14. Scaife, M. and Rogers, Y. External Cognition: How Do Graphical Representations Work. *International Journal of Human-Computer Studies.* (1996), 185-213.
15. Wright, H. Brodlie, K. and Brown M. The Dataflow Visualization Pipeline as a Problem-Solving Environment. *Virtual Environments and Scientific Visualization.* (1996), 267-276.

AUTHOR BIOGRAPHY

Dennis Groth has a B.S. in Computer Science from Loyola University of Chicago and a Ph.D. in Computer Science from Indiana University. His research focuses on the intersection of databases, visualization and interaction. He is an Assistant Professor at the Indiana University School of Informatics in Bloomington, Indiana.

Physical WorkPace: a case study on exploiting context histories in personal healthcare domain

Yuechen Qian
Department of Industrial Design
Eindhoven University of Technology
5600 MB Eindhoven, The Netherlands
y.qian@tue.nl

ABSTRACT
In the Physical WorkPace project we research into how smart rooms will help the user to be more aware of their daily use of computers, and eventually contribute to the reduction or even prevention of physical discomfort and computer-addiction. In the paper we describe our current progress in this project on the design of an everyday object that collects context histories of user activities. We also present some preliminary results on inferring activities from context histories. We believe that our experimental approach towards the exploitation of context histories will motivate a lot of warm discussion in this area.

Keywords
Contexts, context histories, smart environments, pervasive computing, personal care

INTRODUCTION
As more and more work, education, entertainment are carried out in front of computers, people may become victims of Repetitive Stain Injury (also known as RSI, OOS and Carpal Tunnel Syndrome) caused by improper use of computer keyboards and mice. Software tools like WorkPace [1] can help the user to have more frequent breaks and exercise at work. Usually such tools are installed on the computers that the user has access to. Unfortunately those tools can not provide continuous and coherent break suggestions when the user uses computers at different locations, even though the computers are possibly interconnected.

We believe that *context histories*, which contain historical information on user and system activities in smart environments, have great potential for enhancing user experiences in smart rooms, especially on enhancing person healthcare experience. Anti-RSI software tools usually monitor only the duration and frequency of the use of keyboards and mice. With the profusion of sensors, smart rooms could provide more accurate information of the actual work rhythm of the user. Furthermore, existing anti-RSI software does not support cross-device profile exchanges. With the exchange of context histories of the user's work activities across physical boundaries, context histories can be synthesized at any place whenever needed and the joint context histories thus could provide continuous and coherent pictures of daily activities of the user and more sensible healthcare advices could be recommended based on this to the user.

PROJECT TOPICS
In view of ambient intelligence [2], smart rooms can perceive ongoing activities of the user, adapt system behaviors in ritual tasks, and anticipate the needs of the user and act accordingly. Taking this as the premise of our research, we investigate in the "Physical WorkPace" project on how smart environments could help the user to become more aware of their use of computers and regulate it, at work and at home.

In the Physical WorkPace project, we investigate on the follow research topics.

- Design of everyday objects that collect context histories of user activities.
- Algorithms for inferring activities from context history.
- Contextual feedback.
- Exchange and synthesis of context histories.

Exploiting context histories of the user will raise the discussion on privacy, trust, and other social and cultural issues. The results of the discussion may have impact on system design, especially on security and authentication mechanisms. In our project, however, we purposely focus ourselves on technological challenges, due to the experimental nature of our project.

Collecting Context Histories
In smart rooms, sensors are embedded into architecture, furniture and appliances in many ways. In our project, we will look into how sensors could be embedded into or become daily objects. In our point of view, tangibility of

everyday objects will contribute to the acceptance of sensors in their rooms by end users.

Inferring Activities

The data collected by sensors in smart rooms should be used to derive accurate information on user activities. In our projects we are looking for minimal solutions to presence detection of the user at his/her work space. Since we believe smartness will eventual be integrated into home and office environments, minimal resource requirements on sensors and computing power will make our system not only easy to install and maintain, but also affordable to use.

Contextual Feedback

In smart rooms, system behaviors may well be derived by the analysis of context histories, even without direct user interaction. Perceivable system status, thus, is not always the feedback to specific user action. Therefore, it is necessary for the user to understand why the system works in this/that way, or why the system recommends this/that. Anti-RSI software tools suggest breaks at work, based on short-term and long-run user activities. In our project, we will study the techniques, in addition to those used by anti-RSI software, that effectively provide context (history) related feedback.

Exchange and Synthesis of Context Histories

Context histories are gathered at different places and also at different times. To obtain integral and coherent knowledge of user activities, context histories should be exchanged and synthesized whenever needed. In our project, we will research into the synthesis of histories that record user activities at different circumstances (time and location).

In this paper we will present our design and prototyping work on a context history collecting device and some preliminary analysis of gathered data.

PROTOTYPING

At this stage, a device that can detect and collect user presence at work has been designed and prototyped, called *mPacer*. Major functionalities of mPacer are as follows.

- Detect and record user presence.
- Provide break suggestions.

mPacer first detects any movement within its range (about 100cm). It sends a distance reading to a nearby computer via serial communication. The computer records the reading, analyzes the ongoing activity, and sends proper feedback to mPacer. mPacer then displays the feedback.

Since our primary objective is to enhance the user's awareness of his/her daily use of computers, we divide a person's work time into two status: *at screen* and *off screen*, which stand for "using the computer" and "not using the computer" respectively.

In each working status, feedback was designed to indicate whether the user works with proper breaks, as illustrated in Figure 1. Initially, mPacer displays a neutral smiley, indicating no user activities around the computer. When the user starts to use his/her work, mPacer gives a sign signal, a big smile as shown in Figure 2. Similar approaches were used in other intelligent interfaces [3, 4]. When the user keeps on working without breaks, mPacer will give the user a warning signal by displaying a sadness smiley. When the user moves away from the computer, mPacer will display a resting smiley, telling the user to have a good rest and do some exercises. mPacer returns to its neural state when the resting time elapses. In our experiment we define the resting time to be one fifth of the duration of the last work period.

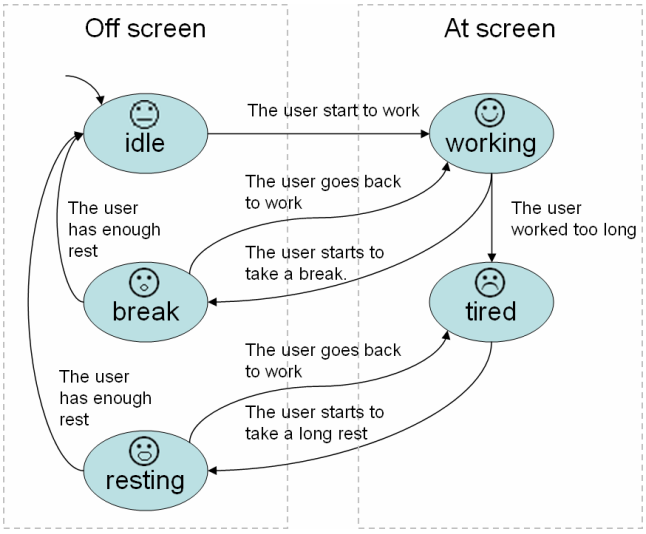


Figure 1 This state transition diagram illustrates the different status (feedback) and in-between transitions of mPacer.

Figure 2 mPacer can detect the presence of a user and provide emotional expressions at the different stages during work.

In our experiment, we use one microcontroller (PIC16F877) from Microchip, one long-distance measuring sensor (GP2Y0A02YK) from Sharp, and a matrix of LEDs for displaying. Together with additional accessory components, those components are integrated into a plastic case, 10cm x 10cm x 6cm (WxHxD). With better prototyping techniques, such devices could be made much smaller.

Currently our mPacer has an office-liked look-and-feel, as illustrated in Figure 2. The material, color and shape of mPacer were chosen to be as neutral as possible. For home environments, the look-and-feel of mPacer will certainly be adjusted accordingly to make it more personal and better suited for the home atmosphere.

INITIAL ANALYSIS

With our prototype device, we invited a colleague from our department for a case study. We put our prototyping mPacer on his desk and to the left of his computer. When the user sat on his chair with rollers, mPacer was in height between the chest and the belly of the user. We asked the user to work as usual and left our device there to run for 30 minutes. Figure 3 shows the sample data collected by mPacer. In the figure, the horizontal axis is the timeline and the vertical axis is the distance between the user and his computer. The closer to the computer the user is, the higher the reading appears to be.

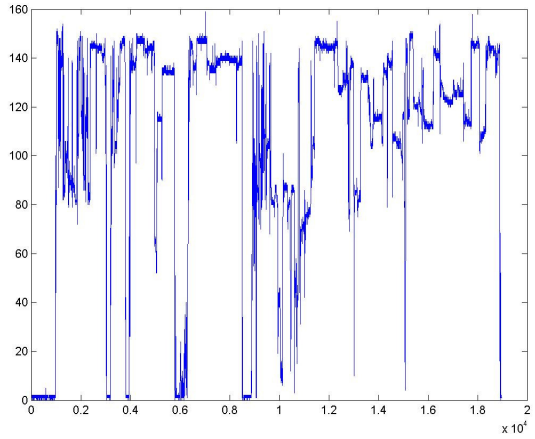


Figure 3 In one case study, user presence at his work was recorded by mPacer. mPacer sampled the distance between the user and his computer for 30 minutes at the frequency 10HZ.

Right after the test, we asked the user to recall his activities during the test. Little matching was found when we compared the activity description by the user with the reading history by mPacer. The main problem there was the activity description made by the user after the study was not detailed enough. To obtain a better idea about the user's activity, we showed the user the reading history by mPacer. All of sudden, the user recognized something and started to explain his activities in great detail!

Although we have doubts about whether the user would still be able to recall his activities in a longer user test, our initial case study does show that our simple and cheap solution to context collecting devices, namely mPacer, is indeed able to collect useful context histories.

Furthermore, we did find the need to create multiple streams of context histories and to synthesize them to obtain accurate information on user activities. Figure 4 shows what the mPacer observed in the first 10 minutes in the study. As can be seen in the figure, initially the readings are pretty low, indicating the user was away from his computer. Then after about 100 seconds, it seems that the user started to work in front of his computer. At this stage, there were a lot of fluctuations, probably showing that the user was cleaning up his desktop and arranging his keyboard and mouse. After about 250 seconds, the signal remains stable for 50 seconds. Without second stream of information, it is very difficult to infer the actual user activities between the 100th and the 250th seconds.

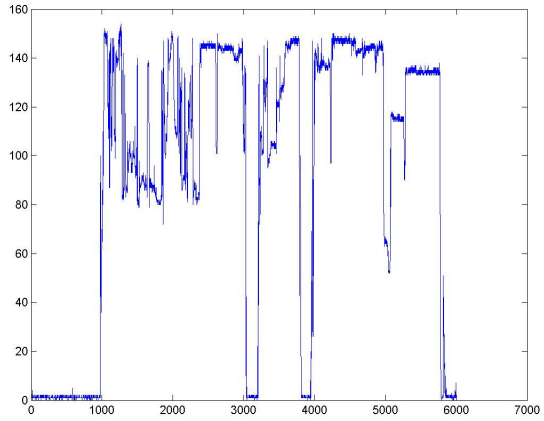


Figure 4 These are the 6000 samples of the first 10 minutes of the case study described in Figure 3.

Overall, the user was very enthusiastic about mPacer and its practical use. He likes the use of emotional expressions in break suggestion, and is in favor of mPacer's clock-liked look and feel. mPacer has been visited by many students and colleagues in our department in a less formal manners and their enthusiasm on mPacer has been recognized as well.

FUTURE WORK

In the rest of the Physical WorkPace project, the following issues will be addressed.

- We will design a software tool to synthesize activity information collected by mPacer and that done by the WorkPace software. Next we will carry out another case study, in which a video camera can be setup to record actual user activities. The synthesized context history will be checked against the taped information on the

video. We expect to find a consistent matching between the two.

- At this moment inferring activities like being at work is mainly achieved by comparing the current reading with a reference reading. We aim to derive finer activity information via signal processing and pattern recognition.
- mPacer currently provides break suggestions when the user works longer than 10 minutes. Also the minimal duration of a rest is fixed at one fifth of the duration of the past work period. Learning algorithms will be implemented to make mPacer be able learn the favorite work rhythm that the user is used to.
- The current feedback system of mPacer may cause some confusion to the user. In our case study, the user told us his feeling that time passed by fast when he concentrated on his work. Sometimes he did not think that he had worked that long and had no need to take a break, though suggested by mPacer. Such phenomena have been attributed to "flow" in Csikszentmihalyi's study [5] and recently reevaluated in computer games [6]. We are going to investigate whether the provision of contextual feedforward and feedback in mPacer will help the user to escape from the flow mode.
- We will also look into technological solutions towards the exchange of context histories between devices across smart rooms. To enable data exchange, we will propose a markup language in XML for describing context histories. To make context histories available for applications in/across smart rooms, context histories management services will be designed and prototyped.

REFERENCES

1. Niche Software Ltd. WorkPace. http://www.workpace.com/.
2. Emile Aarts and Stefano Marzano, The new everyday; views on ambient intelligence, Koninklijke Philips Electronics N.V. 010 Publishers, Rotterdam. 2003.
3. A.J.N. van Breemen, K. Crucq, B.J.A. Kröse, M. Nuttin, J.M. Porta, E. Demeester, A User-Interface Robot for Ambience Intelligent Environments, Proceedings of the International Conference on Advances on Service Robots, ASER, Bardolino, Italia, Published by Fraumhofer IRB Verlag, ISBN 3-8167-6268-9, pp. 132-139, 2003
4. A.J.N. van Breemen & C. Bartneck, An Emotional InterFace for a Music Gathering Application, 2003 International Conference on Intelligent User Interfaces / Philips UI Conference 2002.
5. M. Csikszentmihalyi (1992) Flow: The Psychology of Happiness, London: Rider.
6. Loe M. G. Feijs, Peter Peters, Berry Eggen: Size Variation and Flow Experience of Physical Game Support Objects. ICEC 2004: 283-295

Author Biography

Yuechen Qian is an assistant professor in the Department of Industrial Design at Eindhoven University of Technology. His current research interests include intelligent rooms, context-aware computing, user-system interaction, and digital assets management. He received his B.Sc. and M.Sc. in Computer Science from Nanjing University (China) in 1996 and 1998, respectively. He did his doctoral study on distributed image management at Philips Research the Netherlands and obtained his Ph.D. from Eindhoven University of Technology (the Netherlands) in 2004.

Share Aware: An interactive Pervasive System

To Promote Awareness of Workstation Ergomonics

Anurag Sehgal Interaction Designer
Interaction Design Institute Ivrea Torino, Italy
a.sehgal@interaction-ivrea.it

Patray Wing Lam Lui Interaction Designer
Interaction Design Institute Ivrea Torino, Italy
p.lui@interaction-ivrea.it

ABSTRACT

Workstation ergonomic problems have become a major health issue in the office. Sitting in the same place for a long time and improper sitting postures while using computers can lead to a wide range of computer-related injuries.

We introduce a system promoting workstation health called 'Share Aware'. 'Share Aware' is designed to make people aware of how to prevent computer-related injuries and help them build up correct long term computer-using habits.

The interactive system consists of a seat cover with embedded sensors which track user posture and sitting time with a screen-based application that represents the data in two modes: intrusive or discreet. Real-time feedback, data log of sitting conditions, and customizability are keywords.

This paper discusses the user-centred design and development of the system and how its features benefit the user. These features include context histories, ambient displays, interesting visualisations and self-evaluations.

1. INTRODUCTION: MISSION

115 million days were lost from work due to back pains by computer related work in Britain 1994-95. (Department of Social Security) "After the workstation is built correctly (this is the easy part) the real issue in the office is correct behavior". (Eyal Levy, Ergonomics Researcher)

The interactive system consists of a seat cover with three embedded sensors tracking user posture and sitting time with a screen-based application that represents the data in two modes: intrusive or discreet. For adolescents, a wrong posture while using computers negatively and irreversibly affect body growth. Companies spend an increasing amount on injury compensation claims and the medical costs of occupational illness caused by computers. For workers, work performance is reduced by the physical injuries, which could also lead to work stress. In some cases, RSI (Repetitive Strain Injury) could disable work abilities permanently.

Instead of monitoring computer-using habits, we aim to trigger users awareness and self-consciousness through the use of real-time feedback and reflection of context history.

Figure 1: Shows the postures of many people from our user research. None of them demonstrated correct sitting postures.

2. EXISTING SOLUTIONS

Ergonomic Products

Existing ergonomic chair's have two main functions: automatic adjustment to the human body, or alerting users by embedded alarms. The form of the chair limits its use to be in locations, however, a sensor-embedded seat cover works on any chair design and together with the application can satisfy the needs of users in any work environment that requires an extensive use of computers. Besides, the multi-sensor-embedded ergonomic chair to a certain extent takes away the users control which is a negative effect to user. Therefore, we aim to give control to the user and use subtle hints.

An Ergonomic keyboard aims only at improving hand position. But sitting postures and sitting time are also fundamental factors in computer-related injuries. **Software**

CARPE '04 New York, New York USA

Applications

There are screen-based applications that are preset by the user to alert him or her after a certain duration of time. However they do not account for posture or the fact that the user might not be working at all. Also, such interruption might come at inconvenient times like the presentations. Other applications allow users to take breaks from the computer through peer pressure or surveillance. One application, for instance tells you if your friend has gone for a coffee break and urges you to do the same. We think such solutions invade privacy even though they are setup with the permission of other users because there are moments when one would not like to disclose one's working habits.

3. OUR RESEARCH

Our Goal was to design technologies that help solve problems caused by technologies. After extensive research we focused on workstation ergonomics, an increasingly serious problem in industrialised countries. We then looked at people's careers, lifestyles, technological objects, interfaces, timings, exercises, body parts etc. The following sections describe our Research topics.

Figure 2: Shows our User Testing done at an office.

Figure 3: The user testing helped us iterate our design and come up with a more meaningful solution.

Actors Computers and its various parts, the human body, human postures, health exercises, human senses and possible feedbacks and displays, offices, office work, interiors and workstations, universities and schools, social networks in these locations.

Possible Target Groups Schools, universities, factories, private computer learning centers, typical office, call centers, publishing companies

Chosen Target group The office worker.

Types of physical computer-related injuries We looked at the most common physical problems under Workstation Injuries: Eye Strain, Neck Pain, Carpel Tunnel Syndrome, Triggling Fingers, Back Pain

Correct habit of using computer We looked at what constitutes healthy ways of using computers as suggested by medical experts. We found three areas very relevant in our research: sitting posture, sufficient break, computer positioning

Reasons of having the incorrect habits We then did desktop research and interviewed many people to understand why they do not observe a healthier routine while working on computers. We found the listed factors most common responses were: Cant remember, lack of guidance, ignorant.

Opinions from experts We wrote to ergonomic experts and therapists in leading semiconductor companies with large budgets for research in workstation injuries and their prevention. In particular, the experts mentioned that besides having an ergonomically safe environment, it is more important to help workers develop good ergonomically safe working habits.

Existing tools to prevent or cure computer-related injuries We looked at products and services that presently try to the prevent or cure workstation injuries. We made good use of their findings and learned from the mistakes that they made when designing our own system.

User Test We performed user tests with iterated prototypes at a mobile game design consultancy. Ribes Informatica, Via Jervis 6, Ivrea(To) Italy. 23rd March 2004. Two sets of prototypes were set up in the office for two users. After the set-up, the users worked as usual. Their screens actively displayed their sitting conditions and sitting time throughout the day. Their posture history was being recorded in a text file. At the end of the day we carried out interviews to take reactions, feedback and criticism. The users were shown the text files with their posture history which told a lot about how they worked. They found the real-time feedback was able to make them aware of their sitting postures and thus reminded them to correct these immediately.

Stake holders To understand the business plan we identified some stakeholders who might take particular interest in our service. Organisations: offices, government, health insurance companies, labour organisations, ergonomic consultancies. Individuals: Workers, office executives, ergonomic consultants, physicians, friends, family.

4. OUR PROPOSED SOLUTION: HOW IT WORKS

The design consists of a sensor seat cover and a screen-based application. The sitting conditions and sitting time

will be subtly suggested to the user by a real-time graphical display on the computer screen. The display has two modes: intrusive and discreet .

The application can be customised according to the preferences of users. For instance, they can personalise the application to have a take-a-break alert for every hour they sit.

Data regarding the users sitting habits detected by the sensor seat cover are collected and visualised.

Figure 4: Shows the various components of our system that helps the user evaluate his workstation habits.

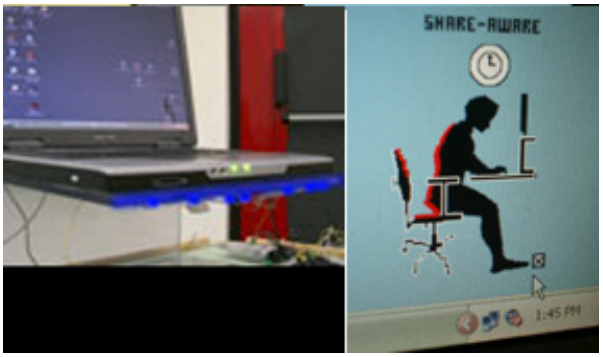


Figure 5: Shows the comparison between the ambient display hidden in the laptop stand and the screen based display in the corner of the screen.

Share-Aware users are given hints about of their posture and duration through with a graphical display on screen. The display takes only a few distinctive features to illustrate the sitting posture but does not completely imitating the way the user sits. This is to avoid suggestion of the surveillance a simple but responsive display is sufficient for the effect we seek.

Only three sensors are embedded in the seat cover because we do not intend to monitor the users sitting pattern completes, which might imply to users that they are being monitored. Instead, we work on raising the users awareness as principle, thus, even three sensors work perfectly in such condition.

Earlier we tested offering the user real-time feedback through an ambient display. However, the effect is unsatisfying. While users work at their computers, their concentration is focused only on the screen. Ambient display is not useful unless it is intrusive enough to catch the users attention off the screen. However, being intrusive could be annoying especially when users are concentrated on their work. Thus, this is avoided in our subtle screen-based solution in which the intrusion level can be personalized by the users.

5. ADVANTAGES OF OUR SYSTEM

5.1 Contributions and challenges

Our discussion of pervasive technology and context history focuses on personal preventative healthcare and devices. The discussion is drawn from the evaluation of our product Share-Aware, which is based on a user centered approach. We now discuss the challenges and contribution of using visualization, real-time feedback and context history with reference to the features of our product.

5.2 User experience of temporal displays and visualization

Screen-based display Vs Ambient Display

Discreet mode Vs intrusive mode

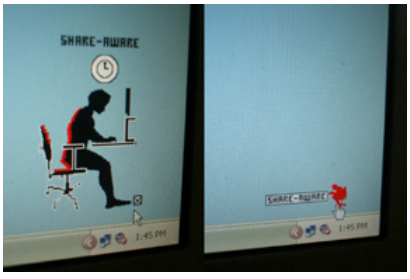


Figure 6: Shows the graphical display in the two modes: Icon mode, a discreet mode and the Display mode, an intrusive mode.

As we noted before the screen-based graphical display can

be illustrated in two modes: discreet and intrusive .The discreet mode is suitable when a users prefer a full screen display of the program they are working at: A very small display at the corner of the screen in the task bar avoids undue intrusion. In contrast, the intrusive mode occupies more screen space and tells users which parts of their posture need to pay attention to and clearly indicates their sitting time. The screen-based graphical display can always sit on top of the other computer applications the user is working at to help gain the user's attention.

Working habits vary from user to user. Having two modes of display addresses the needs of different conditions.

Personalization

Most of the features of the screen-based application can be personalized depending on the users preference.

By default, the sitting time alert is prompted after the user has sat for one hour. The user can configure the time alert to be prompted, for instance, after two hours.

The display size and location of the graphical screen-based display can be customized, as can the duration and type of audio feedback hinting.

We offer these options because we do not intend to change the working habits of users. While using the computer, the user always has full control over the system. They can configure the screen display, volume, screen contrast etc. As they prefer therefore, when we apply the use of pervasive technology, it should not in any way obstruct the users preference nor invade the users freedom.

5.3 Context History

Visualization of History over a Timeline

Figure 7: Shows the data file created storing the users posture and time durations of each posture which is then used to make interesting visualizations of the same information.

Context data history is meaningful to users only when it is visualized and presented properly to them. In Share-Aware, the data is visualized against time. To show users how their sitting patterns improve or vary over a long period. Users can easily identify the possible cause of their computer-related injury, if any, by comparing or analyzing their sitting pattern. Context history is also served to raise the users self-consciousness. The data history stores the user sitting posture and sitting time for each posture which then can be visualized in may interesting ways for different kind of users. Our visualizations shows one such example.

Unlike many existing products Share-Aware does not provide an analysis to the context history. We found that an automated analysis of the data is rarely true and in certain situations might itself cause stress and in turn add to the users injury.We focus instead on accurately visualizing what happened and leave the analysis up to the conscious user who knows their working habits more than any machine could.

Extensive use of Data

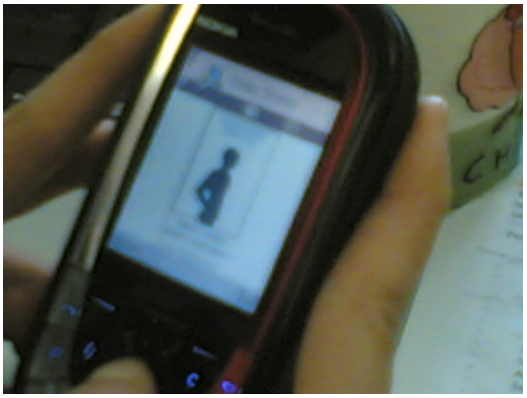


Figure 8: Shows a mobile phone that gives useful feedback to the user by becoming an extension of the system.

Context data history can be used in other mediums. An example in Share-Aware is to offer user a stretch exercise tip through the user's mobile phone in responses to the users sitting problem during the day. It is of the designers interest to interpret the data in different purpose. It is in the designers interest to interpret the data in different ways for various uses.

Sharing Vs Privacy

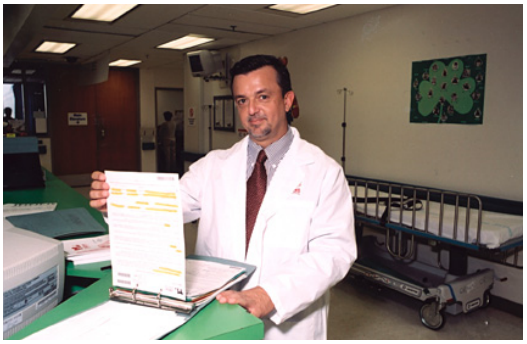


Figure 9: Shows an ergonomic expert who could benefit from such a system.

Though Share-Aware is designed for personal preventative healthcare, we are also concerned about the social impact , earlier we had suggested that our users might share

sitting patterns with each other. We assumed this might create good peer influence amongst users to achieve correct computer-using habits. But users saw this request as an invasion of their privacy. It may be, however, appropriate if the data is presented collectively and compared anonymously. In our context, such collective representation does not serve a very meaningful purpose and therefore this is excluded in our design.

But it is clearly useful to share the data with ergonomics experts if users give their permission.

5.4 Realtime Feedback

Visual: The Share-Aware user is given hints about his or her sitting pattern with real-time feedback. We emphasize hint as we are concerned about the need for subtlety an non-intrusiveness.

Audio: We also found a need for subtle and personalisable sound feedback users are so engrossed in their software that they might miss a discreet visual alert.

In many products a default time fixed alert usually appears as ineffective and even disturbing if it prompts users to take a break when they are just starting to work. Share-Aware, on the other hand, reminds users about their sitting time and postures in response to the particular computer-using habits of different users. Hints are displayed in real time and thus helps users correct themselves immediately. Depending on users, intrusion level preference the application hints to the users any required change in their habit when it matters most to them, that is at that moment itself.

6. AUTHOR BIOS

Anurag Sehgal holds a B.A. in Fashion Design from the National Institute of Fashion Technology, India and a Masters in Interaction Design from the Interaction Design Institute Ivrea, Italy. With his background in fashion design and a graduation project on wearable electronics, Anurag worked on ubiquitous and tangible interface design with The Crossing Project, a research initiative of Xerox PARC and Interaction-Ivrea Explorer Ranjit Makkuni. He has also interned at the ePI lab of Professor Steve Mann at the University of Toronto regarding work on ubiquitous and wearable computing.

Patray Lui received her B.A. in Environmental Design, The Hong Kong Polytechnic University, Hong Kong and a Masters in Interaction Design from the Interaction Design Institute Ivrea, Italy. She worked as an interaction design intern in Philips Design, focusing in multimedia interface. Her current interest is in investigating the interaction quality between a screen-based interface and a physical object.

7. REFERENCES

[1] An introduction to Chiropractic Care. *www.nucca.org.*

[2] backrelief.com *www.backrelief.com.*

[3] Back Injury *www.workcover.com.*

[4] Low back pain amongst nurses in Hong Kong *www.blackwell-synergy.com*

[5] Occupational Neck Pain *www.chiroweb.com*

[6] Positive Health *www.positivehealth.com*

[7] ScreamSaver *www.infinn.com*

[8] Tips for preventing Low Back Pain *www.worksafebc.com*

[9] ZAZCKBACK *www..zackback.com*

[10] Ambient Technology *www.ambientcorp.com*

Integrating History and Activity Theory in Context Aware System Design

Manasawee Kaenampornpan and Eamonn O'Neill

Department of Computer Science,

University of Bath,

Bath, BA2 7AY

United Kingdom

{cspmk, eamonn}@cs.bath.ac.uk

ABSTRACT

In this paper, we describe our context model as a design tool for developing context aware systems. Activity Theory is introduced as a potential approach for identifying and relating the elements that should be taken into account when designing context aware systems. We extend Activity Theory by adding the concept of history to produce the basis for our context modelling.

Keywords

Context, Activity Theory, history, pervasive computing

INTRODUCTION

Two of the factors that can impair the usability of mobile and pervasive systems are increased cognitive load on users attempting to multitask in busy environments, and the restricted input techniques typically available with both mobile and fixed devices. Usability can suffer particularly when there is a need for explicit input. Explicit input is input where the user tells the computer directly (e.g. by command-line, direct manipulation using a GUI, gesture or speech input) what he expects the computer to do, whereas implicit input is an action performed by the user that is not primarily aimed at interacting with a computer system but which such a system understands as input [10]. The need for explicit input may be reduced by increased use of implicit input. Therefore context awareness is an important concept for the usability of pervasive systems as it reduces the need for explicit input by taking advantage of changes in information relating to users, devices and environments. However, the research area of context history [2-4, 7, 11] is quite undeveloped and does not have well-established methods and techniques.

In order to derive principled design methods for developing context aware systems, we require system development processes, tools and techniques that take account of context. We must be able to develop systems that can determine implicitly the data that the user would otherwise enter explicitly. In our research, we have added the concept of history to Activity Theory [9] to provide a design tool to support the designers of context aware systems. Our extension of Activity Theory is used to provide guidance on what elements of context to take into account. It also supports the implementation process and both user- and system-driven adaptability at runtime.

ACTIVITY THEORY

From context classification systems reviewed in [7], researchers have classified context into different elements that have impact on users in performing their activities. Activity Theory is a philosophical framework used to analyse and model human activities. It was developed by the Russian psychologists of the former Soviet Union, Vygotsky, Rubinshtein, Leont'ev and others, beginning in the 1920s [9]. Vygotsky proposed that human actitivities are mediated through tools or instruments; this introduced the first generation of Activity Theory, modelled as a simple triangular structure of Subject-Tool-Object.

Engeström [5] proposed a more comprehensive model of human activity (see Fig. 1). This model was based on the work of the first generation of Activity Theory and on the idea of the general structure of animal activity, consisting of the individual, natural environment and population. Engeström supported the main concept of Activity Theory that individuals' actions are influenced by their socio-cultural context and therefore cannot be understood independently of it [5]. The full triangular structure of human activity that was introduced by Engeström suggests that the relationship between the subject and the community is regulated/mediated by rules and that the relationship between the community and the object is regulated/mediated by a division of labour. Activity Theory maps the relationships amongst the key elements

that it identifies as having an influence on human activity. With Activity Theory, we have a simple standard form for modelling human activity (see Fig. 1).

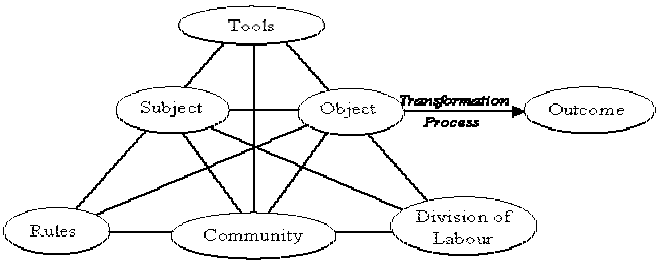


Fig. 1. Full structure of human activity introduced by Engeström et al. [5].

In modelling context for context aware system design purposes, we argue for using a simple standard form to model the aspects of human activity that are associated with key elements of context. Although a simple standard form cannot represent the full richness and complexity of human activity, it does not have to. As humans, we cannot and do not form complete models of other humans' context, especially with regard to their internal goals and intentions. Despite using partial and simplified models, we manage to communicate and collaborate with our fellow humans very effectively and efficiently. From time to time we do get it wrong, for example, misinterpreting another person's intention or meaning. We then invoke repair mechanisms and feed the information generated through this experience into our future models. Since humans manage so well with relatively simple and partial models of other humans' goals and activities, it is both unreasonable and unnecessary to demand more of computer-based context models.

HISTORY

Although Activity Theory captures the key elements of human behaviour, it only captures information about the user's current situation or context and the outcome when the current activity is performed. It does not provide an adequate account of a user's current object or intention, or of the user's past actions and contexts. People often refer to experiences in the past while performing their current activity, using such experiences to guide their current actions. Chalmers [2] notes a range of research that refers to activity as an ongoing temporal process of interpretation. He found significant potential in making more use of the past in context aware system design.

History is a crucially important part of context. A few previous context awareness projects have considered time as context. However, they have typically looked at time simply as current time that can be sensed from the device. For example, they compare current time to the user's timetable and provide support for the user's current task in her timetable [1, 6].

CONTEXT MODEL

Webster's dictionary [8] defines time as "a nonspatial continuum that is measured in terms of events which succeed one another from past through present to future". It defines history as "a treatise presenting systematically related natural phenomena".

Time gives us a means of referencing the occurrence of events; therefore by adding a timeline to the Activity Theory model, we can represent the history element in our context model (see Fig. 2). The timeline includes not just current time, but also past time (which contributes historical elements to the context) and future time (which allows for prediction of users' activities from the current context).

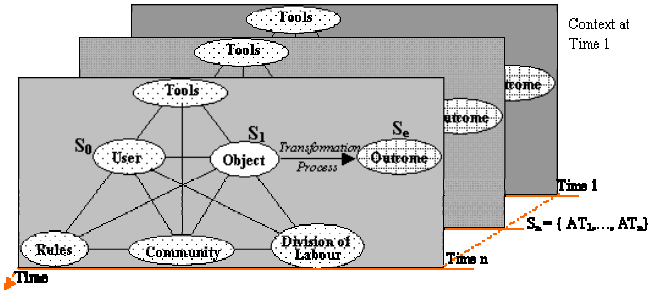


Fig. 2. Extending Activity Theory to represent history.

History is modelled as an abstraction over a set of states in the past. Each past state is represented as an Activity Theory model, which captures the context of activities at that time. This information includes the initial state ($\mathbf{S_0}$), intention or Object in Activity Theory ($\mathbf{S_1}$), and outcome or end-state ($\mathbf{S_e}$) of the activity. The initial state ($\mathbf{S_0}$) includes the current information about user, tools, rules, community and roles. The intention ($\mathbf{S_1}$) models information about the user's current goal, i.e. what the user is trying to achieve. This information about user intention ($\mathbf{S_1}$) can be inferred from the history of context ($\mathbf{S_n}$) and the initial state ($\mathbf{S_0}$). Once the user has performed the activity, we have information about the outcome ($\mathbf{S_e}$). Then, the initial state ($\mathbf{S_0}$), intention ($\mathbf{S_1}$) and outcome ($\mathbf{S_e}$) become part of the history of context and will be used to help infer the user's intention or goal in future situations.

APPLYING THE CONTEXT MODEL

During Design

We propose the context model illustrated in Fig. 2 as a design tool to aid the designer in building an understanding of context. It helps make design issues explicit and forms a basis for design choices. It also encourages the designer to focus on aspects of the system affecting usability.

The context model is used to generate a checklist for the designer to focus on what should be taken into account as context, derived from the elements in the context model. The first step for the designer is to expand the key elements in the context model into sub-elements of information

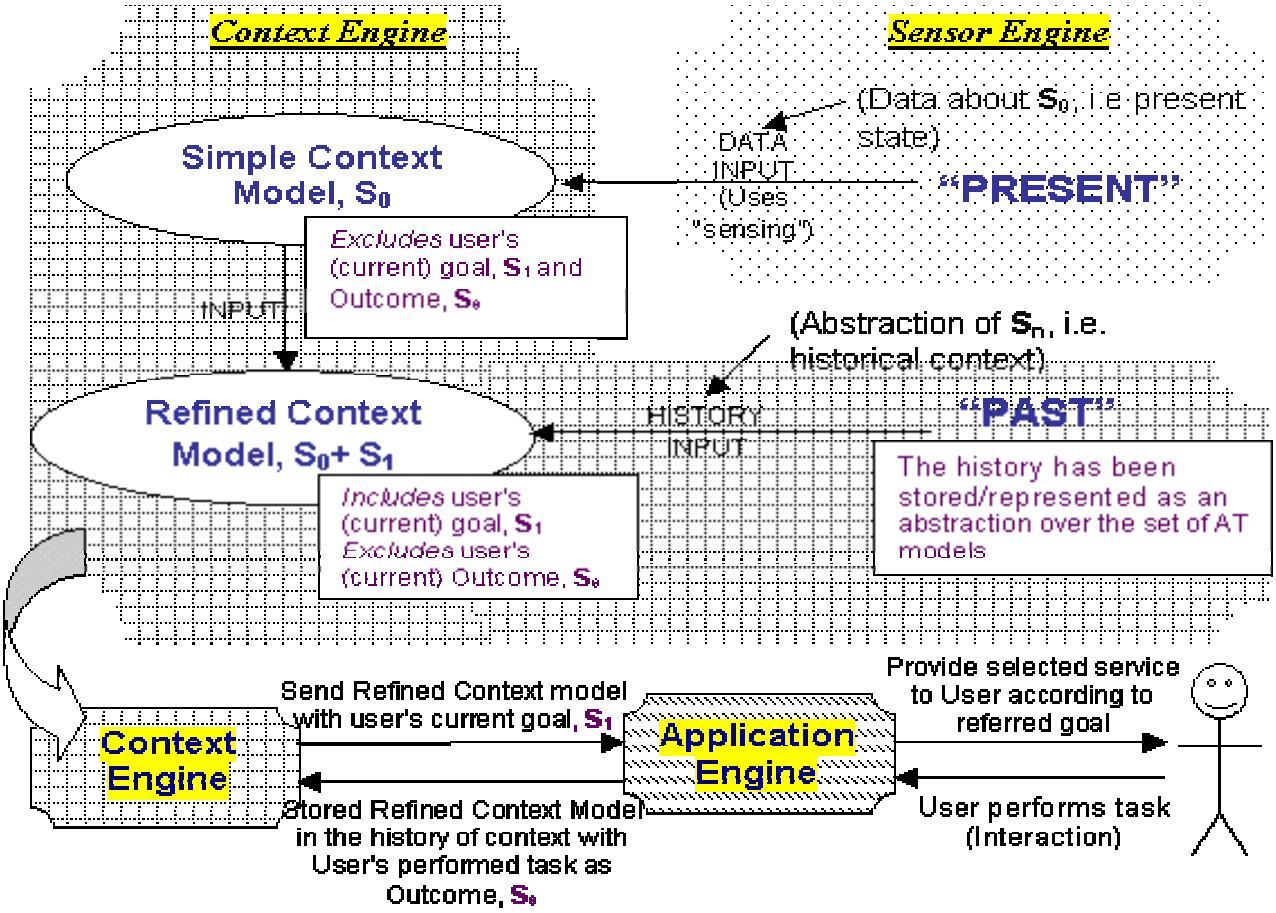


Fig. 3. Information processing in context aware system.

about context, for example information about user's ID, device, location are sub-elements of the User element. The second step is to generate rules to reason about the context according to the relationships between each element in the context model. (These rules are subsequently used to develop a rule-based system, at the implementation stage of the system development process.) The third step for the designer is to check the availability of technologies that can capture the context information in the sub-elements in the checklist.

During Implementation

Fig. 3 illustrates how the system infers the user's current objective, goal or intention. The system first collects information on each element in the context model. This information may be collected from sensors or databases. At this stage, the context model is used as a guide for the system to what types of information to take into account. Secondly, this information is used to model the context of the user's current activity ($\mathbf{S_0}$) based on the rules that the designers have set (and which may subsequently have been adapted at runtime). This simple context model does not include the user's intention or goal ($\mathbf{S_1}$), as we cannot sense such information. Thirdly, the context model references the history ($\mathbf{S_n}$), which records the user's context in performing her activities in the past, in order better to infer the user's current objective. This results in a refined context model that includes a model of the user's current goal. Then the user's current goal is used to identify the appropriate service to support the user's current goal. Moreover, the user's interaction is monitored and recorded as the outcome of the current context model.

Fig. 3 shows that the context model also underlies an implementation architecture that has clear separation between a Sensor Engine (that controls input from different sensors and transforms them into appropriate data input for the context engine), a Context Engine (that matches data input into elements in a context model and infers the user's current goal in the context model by referring to the history of context) and an Application Engine (that selects a service to support the user's current goal by referring to the history of support). Hence, changing sensors or adding/removing application services can easily be done after the system is implemented.

During Run-Time

The system supports adaptation during runtime by both the system and the user (see Fig. 3). For example, the system can downgrade or remove from the reasoning process in

the system the sub-elements of context that have not been used for a certain period of time.

It also supports the user during runtime by giving her a structural understanding about the system; what the system is taking into account as context and how the system reasons about the context. With this understanding of the system, the user is better able to adapt the system according to her requirements. For example, the user can change the rules that have been set in the system when the system does not perform optimally.

BENEFIT OF USING HISTORY

During run-time, the input from sensors may be inaccurate or missing. Fig. 3 shows that input from sensors is processed to fit into the simple context model. The simple context model of the current situation is then used to infer the user's current goal, by referring to the history of context. The process of referring to the history can reduce sensor input problems because the current context is referred to the history as a whole set of context in a model, not as a single input value from the sensor. Therefore, if an input from a sensor is inaccurate or missing but the rest of the values in the context model match the history then we will still get the best-matched current goal.

Similarly, in the application engine, the user's current goal is used to match with the history of services that the application engine has selected to support the user in the past. The history of selected services also holds information about user interaction after the selected service was provided to the user. Therefore the system can use this information to improve performance in selecting the services to support the user's current goal.

CONCLUSIONS

In this research, we considered Activity Theory as an appropriate framework to provide a comprehensive context model that includes the key elements of context that have an influence on a user's diverse activities in a mobile and pervasive computing world. Activity Theory also identifies the relationships between each element in the model so that these relationships may be applied during the development of a context aware system. We identified in this paper that history is important for humans while they are performing their current activity; therefore, we have extended Activity Theory to capture the concept of history in our context model. This model can then be used a design tool for developing context aware systems that reduce the need for explicit input from users. With a design based on a sound model of context and the capacity for runtime adaptability based on past performance and current preferences, such a system will go some way towards achieving the goal of reducing the need for explicit user input and thereby increasing the usability of mobile and pervasive systems in situations of high cognitive load and constrained input devices.

REFERENCES

1. Agarawala, A., Greenberg, S. and Ho, G. The Context-Aware Pill Bottle and Medication Monitor *The Sixth International Conference on Ubiquitous Computing*, Nottingham, England, 2004.
2. Chalmers, M. A Historical View of Context. *The Journal of Collaborative Computing: Computer Supported Cooperative Work, 13* (3-4). 223-247.
3. Chen, G. and Kotz, D. A Survey of Context-Aware Mobile Computing Research, Dartmouth College, Department of Computer Science, UK, 2000.
4. Dey, A.K. and Abowd, G.D. Toward a Better Understanding of Context and Context-Awareness, Georgia Institute of Technology, Atlanta, GA, USA, 1999.
5. Engeström, Y., Miettinen, R. and Punamäki, R.L. (eds.). *Perspectives on Activity Theory*. Cambridge University Press, 1999.
6. Hertzog, P. and Torrens, M., Context-aware mobile assistants for optimal interaction: a prototype for supporting the business traveler. in *The 9th international conference on Intelligent user interface*, (Funchal, Madeira, Portugal, 2004), ACM Press, 256-258.
7. Kaenampornpan, M. and O'Neill, E., Modelling context: an Activity Theory approach. in *Ambient Intelligence: Second European Symposium, EUSAI 2004*, (Eindhoven, The Netherlands, 2004), Springer, 367-374.
8. Merriam-Webster, Merriam-Webster Dictionary. Merriam-Webster, Incorporated, Last Access 17 Feb 2005, (2005) www.m-w.com
9. Raeithel, A. Activity Theory as a Foundation for Design. in Floyd, C., Zllighoven, H., Budde, R. and Keil-Slawick, R. eds. *Software Development and Reality Construction*, Spinger Verlag, 1992.
10. Schmidt, A. Implicit Human Computer Interaction Through Context. *Personal Technologies, 4* (2). 191-199.
11. Schmidt, A., Beigl, M. and Gellersen, H.W. There is More to Context Than Location. *Computers and Graphics, 23* (6). 893-901.

AUTHOR BIOGRAPHIES

Manasawee Kaenampornpan (Jay) received her B. Eng. in Computer Systems Engineering from University of Warwick in 2001 and her M. Sc. in Advanced Computing: Global Computing and Multimedia from University of Bristol in 2002. Currently she is a PhD student in Human Computer Interaction Group at University of Bath under her supervisor, Dr. Eamonn O'Neill.

Eamonn O'Neill is a senior lecturer in the Department of Computer Science at University of Bath. He is a researcher in the field of Human Computer Interaction. He is the leading researcher in "Cityware: Urban design and pervasive system project", commencing in October 2005.

WONDER OBJECTS

Magic and Interactive Storytelling

Tarun Jung Rawat
Interaction Designer
MA Interaction Design
Interaction Design Institute Ivrea, Italy
t.rawat@interaction-ivrea.it
http://jungrawat.com/

ABSTRACT
The inspiration for this project* comes from what we refer to as *cabinets of curiosities*, the forerunners of the modern museum, and it chooses to re-look at museums in context to the role they played as repositories of information and they manner in which they provided this information to their audience.

The first museums of the world were known as 'Wunderkammern' (literally, Wonder Rooms) and 'Wunderkabinette' (Wonder Cabinets), these collections of curiosities, both natural and man-made, offered their viewers a glimpse of the world they had not experienced until then. This project aims to recreate that experience of learning through a sense of discovery and wonder.

It explores interaction design in the context of intuitive and interactive storytelling interfaces, in a museum space, more specifically a 'Museum of Information Technology', displaying some of the famous writing and calculation machines developed by the Olivetti company of Italy during its most productive period.

These interactive storytellers are familiar objects from our everyday world, that we recognize easily, yet they possess an additional hidden layer of information to invoke a feeling of the extraordinary or the magical.

Museums often struggle with the effort of creating an engaging display of their collection of historic objects. This project explores ways in which such dormant inactive entities can be imbued with an animate quality, encouraging the viewer to discover the various hidden layers of information. Through this notion of discovery, and playing on the element of surprise, it seeks to provide a more engaging experience to the museum audience, combining the act of learning with play.

*This paper presents a part of my Masters Thesis Research undertaken at the Interaction Design Institute Ivrea, Italy, in 2004 titled, 'WUNDERDINGE {Wonder Objects} : *Familiar objects as interactive storytellers in a museum space*'. The advisors who assisted me on this project were Britta Boland, Associate Professor at the Interaction Design Institute Ivrea and Alberto Iacovoni from Studio maO, Italy.

Author Keywords
Magic, tangible computing, ubiquitous computing, physical virtual displays, museum spaces, single users, multiple users, interactive objects, interactive displays, easily comprehensible interfaces, active interaction, peripheral awareness, robust technology, surprise, discovery, wonder.

1.0. INTRODUCTION

As computers get smaller, more diverse, and are embedded in the environment around us more frequently than ever, is it possible to extend the inherent language of familiar objects that we instinctively relate to, or know how to interact with at an intuitive level, as a conduit between this physical world that we easily recognize and understand, and the virtual one which is more abstract and ever expanding? Can we create a complimentary relationship between the two by combining the multi-sensory and tangible richness of the former with the dynamic quality of the latter? By adding a layer of digital functionality to these familiar tangible objects, can we bestow upon them a quality of being 'alive' and animate, enriching them even further?

These were some of the questions I asked myself at the very beginning of my study and have frequently touched upon in the course of the development of this project. In this paper I present a set of interactive tools that are designed to provide information about a group of objects on display in a museum setting.

Having always been fascinated by magical objects and the fantastic, the design of these interactive information artifacts carry forward this enthusiasm by exploring the relationship between familiar physical artifacts from our everyday world and the hidden digital layers embedded within, which when revealed, could inspire feelings of surprise and wonder.

When placed within the context of a museum these artifacts function like tangible physical icons of the stories they contain or represent. Together they create an atmosphere, which imparts upon the space and the objects on display, an

animate quality of the living, making the museum a space for wonder and discovery.

As more museums all over the world begin to embrace interactive technologies in a variety of different ways to present their collections, this area offers new opportunities and challenges in re-looking at the museum as a living theatre of memories or a modern day cabinet of curiosities.

Keeping the above in mind, I began my research strongly inspired by the theme of 'magic' and the enthusiasm to explore how notions of magic relate to interaction design in general and interactive objects in particular. I set out to look at those qualities in an interactive artifact that drew parallels with an *object of magic*. When I say an object of magic I refer to the depiction of magical objects and devices as written about in folklore, popular literature, fantasy stories for children and as depicted in films, among other similar sources of inspiration. Objects like the magic wand, the crystal ball, magical instruments of various kinds, talismans and pendants, magical orbs and containers, ornaments and magical clothing, magic mirrors and magical books being a few examples of what one may call 'magical objects'.

Depicted below are some 'magical objects' from the popular Harry Potter film series and the Lord of the Rings film trilogy.

2.0. BACKGROUND REASEARCH

My background research touched upon two key areas. The first being the fields of ubiquitous computing and 'tangible computing', with a strong emphasis on some of the work that has been done by the Tangible Media Group at the MIT Media Laboratory, USA.

The second area of research dealt with understanding the idea of a museum. This meant understanding how the first museums of the world came into being, what did they offer their audience in terms of knowledge and experience, and what do museums today offer us and how - the mediums they use to convey these experiences.

3.0. PROTOTYPES

Presented below are four prototypes that were developed keeping in mind the concept of magic like interfaces as interactive storytellers.

3.1 MagicMirror

Most often information about objects in museums, is provided via textual panels and similar static displays. This piece seeks out other alternative means by which to provide such information in a more intriguing manner. Based on the idea of a 'magic mirror', this solution provides information to the viewer by playing on the elements of surprise and entertainment. By punning on the idea of 'reflections on the typewriter', this piece works in the following manner:

The viewer sees a typewriter, the key object of the display, placed upon a pedestal. It is spotlit (the intensity of the light could vary according to the viewer's proximity to the object, to create a sense of drama). The viewer can also hear sounds of tapping, as if someone is using the machine, giving the object an animate quality. As the viewer approaches closer, the tapping sounds stop, as if the machine were aware of his / her presence, further enhancing this quality of it being *alive*.

Facing the typewriter, a little distance ahead, is a large mirror. This mirror reflects the spotlit typewriter on it pedestal. However, as the viewer comes closer to the mirror, this reflection seems to jump out at him.

What it is

The 'MagicMirror' is essentially a 'two way mirror', with a digital projector placed behind it. Two way mirrors have the capacity to become either reflective or transparent, depending on which of its two faces is receiving more light. The side that receives more light becomes reflective, and the other transparent.

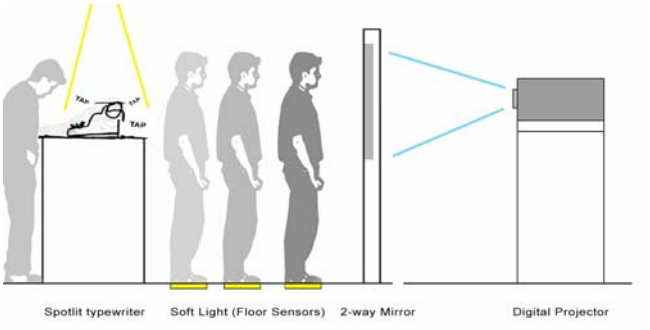
In this way, in its idle state, when the projector behind the mirror projects black (no light), the front surface of the mirror (which is receiving more ambient light) becomes reflective and displays the reflection of the spotlit typewriter on the pedestal. But as the viewer steps closer to the mirror, he activates certain sensors placed on the floor's surface (in this case - keyboard buttons), which trigger visual content on the surface of the mirror, via the projector. The moment the projector begins to display the content, the

mirror transforms from a reflective surface to a transparent backlit projection screen. This happens because, at that instant, the light source from the projector placed behind, is stronger than the ambient light in front of the mirror, hence the apparent transformation. The moment the projector switches back to black, the mirror becomes its reflective self again.

An interesting state in its transformation from a reflective surface to a backlit display is when the viewer steps upon the first floor sensor (keyboard button). The content thus activated, fades in slowly from a low opacity visual, to its full opacity. In the process the mirror is not suddenly transformed into a backlit screen, but instead its reflective quality fades away slowly, to reveal another layer, fading in.

In this way, the 'MagicMirror' becomes a dynamic content display system that uses the proximity and position of the viewer to generate and display content, in an unconventional manner.

For the sake of demonstration, a rough prototype was rigged up using a regular piece of glass in lieu of a two way mirror, a digital projector placed behind it, and keyboard buttons placed upon the floor as the proximity and position sensors.



3.2 WhisperingTable

The second idea, which explores the notion of providing information by inviting the viewer to interact with an object to discover its hidden layers of information, is the 'WhisperingTable'. Inspired by the notion of wonder cabinets and *shadow boxes*, the 'WhisperingTable' is, quite literally, a 'table of content'.

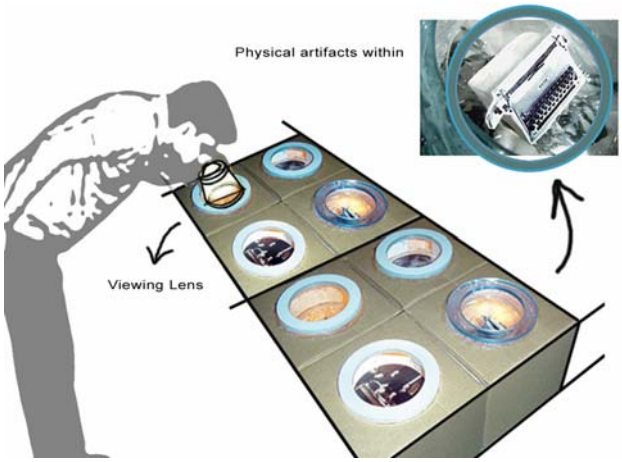
Like the first idea where the typewriter seems to be *alive*, this installation builds upon the similar theme of the animate object.

The 'WhisperingTable' is a table with small peep-holes on its surface. It is placed in a relatively dimly lit part of the space. In its idle state, the viewer can see a flicker of lights and hear a murmur of sounds, emanating from the table. This entices the viewer to approach the table. As the viewer approaches, the flickering stops as do the sounds. The table is seemingly aware of the viewer's presence.

What it is
The 'WhisperingTable' is essentially a hollow table with peep-holes on its surface. These peep-holes function as windows to the individual compartments below. These compartments are illuminated from within and the level of illumination is variable (like the effect of a light dimmer). In its idle state, these lights change their state from full illumination to a dim state, creating a flicker effect. Similarly, in the idle state, the audio content of the table, begins to play randomly and at a low volume, creating a murmuring / whispering effect, to draw the viewer to the table.

Each individual compartment houses artifacts and small digital screens, that when triggered launch contextual audio / video content. This content is triggered by the viewer placing a viewing lens (provided for on the table-top) upon any one of the peep-holes. The peep-holes, otherwise dimly illuminated, brighten up when the viewing lens is placed upon them, thereby bringing into focus the key peep-hole of the moment, while keeping the content of the others still a mystery, waiting to be unravelled.

For the sake of demonstration, a quick idea sketch was developed using cardboard boxes and keyboard buttons as the pressure sensor triggers.



3.3 InteractiveBook and WallCabinets

This installation has two key components:
a) InteractiveBook (plus fragmented/distributed projections)
b) WallCabinets

a) **InteractiveBook**
A book is an object that is symbolic of information. We have a tacit understanding of how to use it. We know, that to access the information within, we have to open it and flip

through its pages. Based on an earlier prototype called 'The Book of Answers' done by Aparna Rao, a colleague of mine at IDII, the 'InteractiveBook' explores this notion of our intuitive interactions with an ordinary book, to generate content in a not so ordinary manner.

What it is
Upon a spotlit pedestal is placed the key object of the display, a typewriter. Placed at a slightly lower level is a book. This book provides information about the typewriter on display. However it does so in an unusual manner, as the viewer soon discovers. At the first level, its pages provide textual information and printed imagery, as does a conventional book. But at a second level, certain pages once flipped by the viewer, launch contextual video projections onto a fragmented display system, making the 'InteractiveBook' a simple tangible and intuitive interface to access information. It plays on the theme of the 'magical object'.

The video projections take place on a set of papers that are suspended from the ceiling, seemingly floating in space. These papers are placed at varying distances from one another, creating an illusion of depth and producing a fragmented image of the whole. When viewed from the front, the viewer sees the whole image, but when viewed from any other angle, these floating papers appear like illuminated windows, animating the space in the periphery of the key exhibit. This fragmentation is suggestive of different ways of looking at the key object on display, as well creating an atmosphere of drama around it.

b) **WallCabinets**
The second component of this installation are the walls of the space surrounding the typewriter on display, and the 'InteractiveBook'.

Carrying forward the inspiration from *Wunderkabinette* (Wonder Cabinets) and building upon the idea of the 'WhisperingTable' is the concept of the 'WallCabinets' - the wall as an interactive space.

What it is
The walls surrounding the key display are embedded with a matrix of small windows / compartments, each displaying an image or artifact placed within. Some of these compartments have doors with handles (simple interface cues), which the viewer can open to reveal a hidden layer of information, in the form of audio or video content, transforming the entire wall into a large interactive wonder cabinet. The images and artifacts displayed within these compartments are arranged in context to the object on display, to give the viewer a broader sense of the times i.e. events, design directions and popular culture that existed, when this object was designed and produced.

While 'InteractiveBook' provides a micro view of the typewriter on display, the 'WallCabinets', like the *Wunderkammer* (Wonder Room), provide the viewer with a macro view - a glimpse at its larger context.

3.4 WonderObjects

In this third display, set up to present information about another landmark typewriter, the viewer finds a table with a typewriter placed upon it. Video content is projected from above onto the surface of the typewriter, and the surface of the table itself.

In its idle state the visitor sees and hears hands typing on the machine's keyboard. The video has been created in such a fashion that there is a precise one is to one layering of the virtual image upon the actual physical object, transforming the static object into a surreal animate entity.

As the viewer comes closer (via proximity sensing), he triggers another video, and the hands begin to type out a set of instructions which appear on a sheet of blank paper that is inserted in the machine. These are an index to certain 'hotkeys' on the typewriter's keyboard that when pressed, trigger contextual video content, animating the object on display and creating these 'wonder objects' that seem to pulse with life, haunted by an immaterial presence that enables them to tell their stories to us

CONCLUSION

By presenting the examples mentioned above, this paper illustrates ways in which an ordinary artifact in a museum space may be made more interesting to the viewer, by encouraging an active interaction between the viewer and the object on display. The interaction process has been designed with a strong focus on the elements of surprise and discovery, to make the viewer's experience a little like the Wonder Rooms and Wonder Cabinets of the 16th and 17th centuries - an experience of learning by wondering at and wondering about the object on display.

ACKNOWLEDGMENTS

Interaction Design Institute Ivrea for all its support, and the Yeditepe University, Istanbul for providing the opportunity to present this paper and for its publication.

REFERENCES

• DIS2002, *Serious Reflection on Designing Interactive Systems*, Publication of proceedings from Designing Interactive Systems 2002 (June 25-28, 2002, The British Museum, London), ACM Press, 2002.

• Edwards, W.K and Grinter, R. E. *At Home with Ubiquitous Computing: Seven Challenges*, Computer Science Laboratory, Xerox Palo Alto Research Center, California, US.A.
In: G.D Abowd, B. Brumitt, S.A.N. Shafer (Eds.); UBICOMP, 2001, LNCS 2001, pp 256-272, 2001 (copyright – Springer-Verlag Berlin Heidelberg 2001).

• Holmquist, L. E., Redstrom, J. and Ljungstrand, P. *Token-Based Access to Digital Information*.
In: Gellersen, H. W (Ed.); Handheld and Ubiquitous Computing, Lecture Notes in Computer Science No. 1707, Springer-Verlag, 1999, pp 234-245.

• Ishii, H. and Ullmer, B. *Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms.* In Proceedings of CHI '97 (March 22-27, 1997), ACM Press, 1997.

• Ishii, H. and Ullmer, B. *mediaBlocks: Tangible Interfaces for Online Media*, Published in the Conference Abstracts of CHI'99 (May 15-20, 1999), ACM Press, 1999.

• Ljungstrand, P. , Redström, J. and Holmquist, L. E. *WebStickers: Using Physical Tokens to Access, Manage and Share Bookmarks to the Web*.
In Designing Augmented Reality Environments (DARE'2000), Elsinore, Denmark, April 12-14, ACM Press, 2000.

• Marchak, F. M. *The Magic of Visual Interaction Design*, SIGCHI Bulletin Volume 32, Number 2, April 2000.

• Poynor, R. *The Hand That Rocks the Cradle*, I.D. Magazine, May/June 1995, pp 60-65.

• Rosenblum, M. and Macedonia, M. *The MagicBook - Moving Seamlessly between Reality and Virtuality*, IEEE Computer Graphics and Applications, May/June 2001.

• Smith, M., Davenport, D. and Hwa, H. *AURA: A Mobile Platform for Objects and Location Annotation*, Published in Adjunct Proceedings of Ubicomp 2003 (October 12-15, 2003), ACM Press, 2003.

• Svanaes, D and Verplank, W. *In Search of Metaphors for Tangible User Interfaces*.

• Tognazzini, B. *Magic and Software Design*, "Principles, Techniques, and Ethics of Stage Magic and Their application to Human Interface Design."
In Proceedings of INTERCHI, 1993 (Amsterdam, the Netherlands, April 24-29, 1993), ACM Press, New York, pp 355-362.

• Ullmer, B. *Models and Mechanisms for Tangible User Interfaces*, MIT Media Lab, Tangible Media Group, Cambridge, MA, USA, 1997.

• Ullmer, B and Ishii, H. *The metaDESK: Models and Prototypes for Tangible User Interfaces*. In the Proceedings of UIST '97, October 14-17, 1997, ACM, 1997.

• Ullmer, B and Ishii, H. *Emerging Frameworks for Tangible User Interfaces*, IBM Systems Journal, Vol 39, Nos. 3 & 4, 2000.

• Want, R. *Remembering Mark Weiser: Chief Technologist, Xerox PARC*, IEEE Personal Communications, February 2000.

• Weiser, M. *The Computer for the 21st Century*, Scientific American, 1995.

• Weiser, M and Brown, J. S. The Coming Age of Calm Technology, Xerox PARC, October 1996.

• Wellner, P. *Interacting with Paper on the Digital Desk*, Communications of the ACM, July 1993.

• Wisneski, C., Ishii, H., Dahley, M. G., Brave, S., Ullmer, B and Yarin, P. *Ambient Displays: Turning Architecturalinto an Interface between People Information*. In the Proceedings of the First International Workshop on Cooperative Buildings (CoBuild '98), February 25-26, 1998, Springer, 1998.

Bio

Tarun Jung Rawat is an Interaction Designer and independent researcher. He holds an MA in Interaction Design from the Interaction Design Institute Ivrea, Italy (2004). His current projects are inspired by issues arising from the realms of ubiquitous computing and more specifically, the area of tangible computing. His projects explore various interaction design/human computer interface related work within this framework. He is presently based in New Delhi, India. Examples of his work may be found at:

http://jungrawat.com/

Exploiting Context Histories in Setting up an e-Home

Johannes Helander
Microsoft Research
1 Microsoft Way
Redmond, WA 98007 USA
+1 425 882 8080
jvh@microsoft.com

ABSTRACT
Turning a home into a seamlessly integrated, yet secure environment, without excessive cost, presents a number of challenges. This paper attempts to draw various solutions together. What must be done for security, can also be exploited in making configuration less tedious.

The home environment is augmented by low-cost invisible computers that let everyday objects communicate and integrate. Embedded XML web services are used as a generic substrate for exchanging information between all classes of devices: simple light switches to complex personal computers. Solid cryptography and a touch based trust establishment protocol allow setting up a secure home completely independently. Finally the human interaction context history is used to heuristically determine how the different devices should interact.

Keywords
Home automation, XML web services, embedded systems security, context histories.

INTRODUCTION

An automated home can make our lives more comfortable and easier, perhaps lengthening the time the aging population can stay independent. Countless homes are already filled with personal computers, music systems, appliances, light dimmers, and security systems. If all these systems could work together the utility of all the devices would improve.

There are two main problems in the status quo, however. 1) The systems are difficult to set up and it is impossible to make them work together. Beside the physical connections, that often could be replaced by wireless connections, the protocols spoken are proprietary and application dependent. The user is relegated to archaic switches and menus to tell the systems what to do. A seamlessly integrated home is currently achievable only for millionaires with professional installation crews. 2) The systems are insecure and compromise the privacy of the inhabitant. By installing automation systems the owner of the home ends up paying for losing control of information pertinent to their sanctum. The use of wireless connections only makes the situation worse.

The author claims that interoperation has to be built right into the basic functionality of the system. Security and privacy is not an afterthought and should not make the system another order of magnitude more complex and difficult to set up. The user experience must be intuitive and natural, preferably completely invisibly result from the unavoidable physical installation without further configuration steps.

This paper proposes a physical touch based functional and trust establishment mechanism that exploits a contextual history of human interaction. Interoperation is achieved through the use of XML Web Services that run on low cost microcontrollers. A public key cryptography based trust manager achieves security without external trust authorities. The trust setup is based in physical touch; as a side effect a context history is created. This context history is exploited to determine functional relations between the devices. A single touch per device is thus all that is needed, and this applies potentially to all the devices in the home

EMBEDDED XML WEB SERVICES

Web services were conceived to solve the e-business interoperation problem. The same problem is very pressing in a home environment. The author has shown in [1] that web services can also perform on low-cost microcontrollers. Entertainment content streaming and privacy add new issues, the use of web services in addressing these issues have been explored in [2] and [3].

TRUST ESTABLISHMENT

A home should be totally controlled by the owners. The owners should also be able to set up their home without outside assistance or authority. This is achieved in [2] by using the Resurrecting Duckling Protocol [5]. It works by defining one device to be the authority—the mother. New devices believe the first other device they see is their mother unless they already have one. The first contact is thus critical and a touch (or proximity) based channel is used. The physical touch signifies a human intent and physical access to the mother device. Each device is identified by a certificate that contains the public key of the device, delivered over the physical touch channel. The

certificates are signed by the mother with its private key. The mother certificate is received on the same physical touch channel. All later communication can happen on a regular wireless or other public data link.

There is no need for central certificate authority outside the home. [2] shows how independent authorities can federate to manage mutual partial trust.

CREATION OF THE CONTEXT HISTORY

As the creation of the home trust domain and admitting devices into it involves human interaction, the precise pattern of that interaction can be recorded. A single instance of interaction is a context event.

When new devices are brought into the home, they are touched by the mother, e.g. a smart watch, to make them part of the family. It is also possible to touch a device at other times at will.

REPRESENTING CONTEXT EVENTS

A context event, like any data, is represented as an XML fragment. The fragments are collected by the watch and can be sent to any interested and trusted parties based on event subscription.

```
<interaction time="2005-02-21T18:25:00Z" type="touch">
  <function type="light" subtype="torchier" power="100" unit="watt"/>
  <location="floor" height="1.5" unit="meter"/>
  <watch-buttons>1 2</watch-buttons>
  <contact id="uuid:7796f8ac-ab60-49e5-a2e7-61db77e64096"
           url="http://123.45.67.89/discovery"/>
</interaction>
<interaction time="2005-02-21T18:26:00Z" type="touch">
  <function type="light-switch" subtype="lever" values="on off"/>
  <location="wall" height="2" unit="meter"/>
  <contact id="uuid:1a0e6bd0-806f-4bd8-8e93-c0afd97b1044"
           url="http://234.56.78.90/discovery"/>
 </interaction>
```

The context event contains the type of the device and its functions as well as how to contact it. Beside the time, the event contains the location to the extent known and any available information of what the user did, such as buttons pressed at the time of the interaction. The log is conceptually stored in a distributed data base, where it is available for queries and data mining.

EXPLOITING THE CONTEXT HISTORY

Once a context history is available, the device uses it to determine what it is supposed to do and what other devices it should be associated with. The choice is constrained by the trust domain. Untrusted devices are simply ignored, although federation of trust domains enables limited cross-domain interaction.

The primary source of information is the timestamp in each event. Those events that are close to each other temporally can be assumed to be related. Since the pace of user interaction depends on the speed of the user and on the proximity of the devices, an absolute time difference would be inappropriate. Instead a statistical clustering algorithm is used. First obvious (multiple hours) gaps are used to partition the history into sets. Next every interval is examined and a normal distribution is calculated for a set. The set is then separated into multiple subsets by picking a threshold value for the deviation. The largest deviation gaps are used as cutoff points to separate events into separate subsets. The threshold is progressively lowered and too large sets are separated into two. This is done until every set is of reasonable size (<= 5). Finally it is determined that each of the smaller sets is a separate cluster of related functions.

The clusters are examined for compatible functions such as a light and a light-switch. Compatible functions are then linked together. Simply put, the relative temporal proximity of two compatible devices determines their functional relationship. If no compatible functions are found, the proximity requirement is loosened by backtracking the set splitting until some useful relationships are found.

When location is known, it used together with the temporal proximity to determine overall proximity by mixing temporal and spatial proximity together with heuristically determined weights.

The result of the functional (partitional) clustering is that the light-switch ends up controlling the light given that they were both touched in a reasonably close time span.

If desired, the user can control the process to indicate that the clustering should be split at a given point. For example, pressing button 2 on the watch while touching a device could signal that the current device has nothing to do with the previous devices. This could be done when moving from one room to another without a break in between.

RESULTS

The secure embedded web services and the resurrecting duckling protocol that provides the context events were implemented on a low-end ARM microcontroller [6]. Similar single-chip low-power computers are currently available for roughly $5.

We evaluate the feasibility of the software and the security protocol with measurements. Table 1 shows that the entire software can run on a computer that has 256KB ROM and 32KB RAM. This is available on modern microcontrollers of interest.

Files	ROM	Static RAM	Heap	Stack	Total RAM
BASE	24,676	1,940	2,837		2,777
DRIVERS	11,464	332	896	2,288	3,516
TCP/IP	77,024	3,424	2,648	3,400	9,472
XML	7,860	16	88		104
SOAP	29,504	280	996	4,320	5,596
SECProto	14,180	604	1,848	2,648	5,100
AES	16,532	8			8
RSA	9,784	28	24		52
SHA1	5,436	8			8
C-Library	7,620	12			12
TOTAL	**204,080**	6,652	9,337	12,656	**28,645**

Table 1: Footprint (arm - in bytes) at peak usage

We evaluate whether the solid cryptography is feasible on low-cost devices. Table 2 reveals that the two significant costs are key generation and RSA private key operations. The former only needs to be done, and can be primed at the factory or on the way home. RSA private key operations are needed for certificate signing and key exchange. Each need to be done only once but cannot be done before the device was touched. Luckily the certificate does not have to be signed while touching so the interaction itself is quick.

After touching a device but before two devices can communicate, two RSA private key operations must be done in sequence. This means that it takes almost half a minute before the newly associated devices can communicate to each other, making immediate feedback problematic. Further work will investigate cutting down this delay perhaps driven by the mother device.

The clustering algorithm does not contain any complicated math and can be completely calculated using fixed point integer arithmetic in linear time. This makes it suitable for microcontroller use as compared to more elaborate and sophisticated schemes such as [4] that uses Markov models and Bayesian networks, where the clustering computation itself could exceed the available computational capabilities. The simple clustering algorithm presented here also has the advantage of working with little stored history, yet sufficiently addresses the problem requirements.

FURTHER WORK

User studies would be beneficial in determining the best clustering of events and for tuning the algorithm and for verifying that the results are those expected by most users. Further experimentation with exploiting other known parameters, such as partial locations, might also yield interesting results.

CONCLUSION

It is possible to create an interoperable home automation architecture that is both easy to use and affordable without compromising privacy. Data mining of context histories is a viable way of extracting information from interactions that are already necessary for other reasons. This information, when combined with other known information provide enough context to avoid tedious configuration menus and complicated setup steps. The preliminary work presented in this paper shows that this is possible but user studies are needed to determine whether the heuristic is strong enough to produce results intuitive to most people in variable environments.

REFERENCES

1. Forin, A., Helander, J., Pham, P., and Rajendiran, J.: Component Based Invisible Computing, *3rd IEEE/IEE Real-Time Embedded Systems Workshop* (London, December 2001).
2. Helander, J. and Xiong, Y.: Secure Web Services for Low-cost Devices, *8th IEEE International Symposium on Object-oriented Real-time distributed Computing* (Seattle, May 2005).
3. Helander, J. and Sigurdsson S.: Self-Tuning Planned Actions: Time to Make Real-Time SOAP Real, *8th IEEE International Symposium on Object-oriented Real-time distributed Computing* (Seattle, May 2005).
4. Moore, D, Essa, I., and Hayes, M.: Exploiting Human Actions and Object Context for Recognition Tasks, *7th IEEE International Conference on Computer Vision* (Corfu Greece, 1999).
5. Stajano, F. and Anderson, R. The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks *LNCS 1796*, Springer-Verlag, 1999.
6. AT91M63200 Summary, AT91 ARM Thumb MCU, http://www.atmel.com/dyn/resources/prod_documents/1028S.PDF
7. The embedded web services implementation is available at http://research.microsoft.com/invisible/

Algorithm	Operation		Latency on a 25 MHz ARM 7		
			Average	Standard deviation	Per KB
1024-bit RSA	Generate a key pair		290 s	56%	N/A
	Private key	Encrypt/decrypt a block (128 bytes)	12.9 s	<1%	103 s
	Public key	Encrypt/decrypt a block (128 bytes)	0.667 s	<1%	5.34 s
128-bit AES	Encrypt/decrypt a block (16 bytes)		0.254 ms	<1%	16.3 ms
SHA1-HMAC	1024 bytes		79.6 ms	<1%	79.6 ms

Table 2: Speed of cryptographic primitives

Privacy Enhanced Active RFID Tag

Shingo Kinoshita, Miyako Ohkubo, Fumitaka Hoshino, Gembu Morohashi, Osamu Shionoiri, and Atsushi Kanai
NTT Information Sharing Platform Laboratories, NTT Corporation
1-1 Hikari-no-oka, Yokosuka-Shi, Kanagawa 239-0847 Japan
{kinosita, ookubo.miyako, fhoshino, gembu}@isl.ntt.co.jp,
{shionoiri.osamu, kanai.atsushi}@lab.ntt.co.jp

ABSTRACT
In the coming future ubiquitous society, Radio Frequency Identification (RFID) tags will be affixed to every product and person. This technology is anticipated to be a key technology that will be utilized by various ubiquitous services where these tags will be used to identify things and people and will automatically take advantage of contextual information such as location. On the other hand, a problem is arising where the excellent tracking ability of RFID is abused and personal privacy is being violated. This paper clarifies the active tag privacy problem and proposes a method for protecting personal privacy regarding the active RFID tags. In the proposed method, re-encryption technologies are used to make the tag ID variable. Since variable IDs generated from one ID are cannot be linked to one another by third parties, RFID privacy problems based on a fixed ID can be abated. Furthermore, we introduce an active tag prototype that implements the proposed method and evaluated its effectiveness.

Keywords
RFID, active tag, privacy, security, encryption

1. INTRODUCTION

Radio Frequency Identification (RFID), an automatic recognition technology employing wireless communications, has recently drawn much attention. RFID tags, which are electronic tags that employ RFID technologies, can be broadly classified into passive and active types of tags. The passive type does not incorporate a battery and has a short communications range, but the cost is low. Conversely, the active type incorporates a battery and has a long communications range, but the cost is high.

Although these tags differ in terms of the communications range and cost, they are often attached to products and goods as a means of inventory management. Very recently the number of applications has increased where tags are attached to people. A representative example of this type of application of a passive tag is a facility access card, which is often used by businesses. In Japan, this application is not limited to businesses, it is also used for entry cards to exhibitions [1]. The active tag can be used for marketing, for example, it can be used to track the behavior of customers. Furthermore, for the purposes of security and safety, tracking the behavior of kindergarten children [2], monitoring grade school children on their way to and from school [3,4], and locating wandering or missing elderly people have been initiated.

By using this type of bearer tag, the identification of the bearer and contextual information such as location can be easily obtained with relative certainty. In the future anticipated ubiquitous society, this type of high-level identification will be a very basic technology and play an important role.

On the other hand, in the case that the excellent automatic recognition and tracking abilities of the RFID technology are abused, the privacy violation is a problem. There are already various protest movements that target the use of passive tags [5] and the bearer type active tags [6]. Especially in the case of active tags, which have much longer communications range than passive tags, this is a serious problem.

This paper proposes technological countermeasures that resolve the privacy problem related to active tags and introduces a prototype that we developed.

2. ACTIVE TAG SYSTEM

2.1 What's Active Tag

An active tag is an RFID tag that incorporates a battery, and can communicate with a reader that is several tens of meters away (there are tags that can communicate at several hundreds of meters). While passive tags can only respond to an electromagnetic wave signal emitted from a reader, active tags can also spontaneously transmit an ID. There are various types of transmission opportunities such as the very common periodic transmission type, or the unscheduled transmission type such as when there are changes in vibration or temperature or when a button is pushed. In many cases, the ID data comprise several tens of bits.

Generally, systems that employ active tags comprise the tags, a reader, and a server. The tag spontaneously transmits its ID. For example, if the tag is a periodic transmitting type, the tag transmits its ID once every several seconds. When the reader receives the ID, it

notifies the server of the ID via the network, and based on the ID the server executes the target service.

2.2 Active Tag Applications

This section introduces application examples for active tags.

[Behavior tracking of kindergarten children] [2]

Parents or guardians can view their children in kindergarten via the Internet by utilizing the active tags. Active tags are attached to the nametags of the children, and the classrooms and sports grounds are equipped with a reader and a Web camera. Based on this, by accessing the Internet the children can be viewed in real time and in their actual surroundings. The parents or guardians can automatically select video images of their children.

[Monitoring grade school children on their way to and from school] [3,4]

Since the incidences of abduction and brutalization of children as they are on their way to and from school has increased, the application of active tags has been investigated. The backpacks etc. of the children are equipped with a tag and readers are installed along the route to school and at the school gate. When a child passes by a location that is equipped with a reader, the ID is transmitted and the school and the parents or guardians are notified. By using this system, at an early stage the teachers and the parents or guardians can become aware of any abnormalities in the commute to school.

[Monitoring with the aid of cameras]

Through the cooperation of monitoring cameras and an active tag system, if images are recorded at the same time that the ID is received and recorded as metadata, an effective method for investigating criminal offenses becomes possible. For example, it would be very efficient to use the ID of an abducted person as a search key in an image search.

[Promotion and marketing]

In department stores and supermarkets, if customers bear tags, their behavior can be tracked inside the store, and based on their context history such as moving path or purchasing history, the consumer can take part in promotions that are made possible through the Kiosk terminal inside the store.

[Authentication and settlement]

The use of contact-less IC cards for ticket examination in traffic systems has increased, and the system has become very convenient. To advance this concept further, if active tags can be used in authentication, it would even save the trouble of taking out a card. This type of process would become effortless and the level of convenience would increase even more. Of course, being billed for simply coming into close proximity of these readers would be problematic, and an authentication and settlement scheme that prevents illegal acts such as impersonation is needed.

In this way, applications that use active tags have a wide range and have the potential to become the basic identification method for future ubiquitous services.

3. RFID PRIVACY ISSUES

On the other side of this convenient system, there is the increased anxiety caused by privacy violation stemming from automatic identification using the active tags. This section evaluates the threat to privacy that can occur by transmitting an ID, which at most comprises several tens of bits. First, the characteristics of the many currently used active tags are clarified.

- The active tag transmits its ID without the knowledge of the owner. More specifically, the owner does not have to perform an action such as consciously pushing a button as in the case of an immobilizer. The tag periodically and automatically transmits the ID.
- Anyone that possesses a reader can receive the ID.

These two characteristics lead to the consequence that anyone possessing a reader can receive the ID without the owner being aware. Whether or not this idea can actually be connected to the violation of privacy depends on the characteristic of the ID being disclosed as described below.

3.1 Content Privacy

In the case where the ID contains personal information pertaining to the tag bearer or other related information, there is a risk that others can easily obtain this information. For example, the ID could be assigned information such as the gender of the bearer, birth date, zip code, telephone number, employee number, and student number. For the current active tags, since each service can freely determine the information contained in the ID and there are still many immature service providers that have a low level of awareness of the crisis related to the privacy threat, the possibility cannot be denied that this information may naively be included in the ID.

3.2 Location Privacy

Even if personal information is not included in the ID as mentioned above, if the ID is fixed, there is a danger in that the behavior history of the tag bearer can be disclosed to others based on ID tracking. This danger does not stop at simply the physical tracking of locations visited. All kinds of personal information can be obtained by analyzing the types of places visited such as hospitals, schools, and stores.

Obviously, the ID and the bearer must be connected to be effective. Anyone can very easily obtain the ID information of the bearer by simply coming into close proximity to the bearer and reading the ID using a reader. Conversely, for the ID to specify the bearer is comparatively difficult. The degree of difficulty depends on factors such as whether or not a database (DB) exists to connect the ID to the personal information and the strength of the security of the DB.

Furthermore, it is very dependent on the uniqueness of the ID. As mentioned earlier, in the current state, since each service freely determines the contents of the ID, at best only within the service can the uniqueness be guaranteed. More specifically, when considering local, national, and international levels, the possibility is high that duplication will occur. As the degree of duplication increases the connection between the ID and the tag bearer becomes weaker and it becomes more difficult for a privacy problem to occur.

However, in the future, in the process in which the active tag will be developed as a fundamental device in the global ubiquitous society, the tag ID will also be standardized and made globally unique similarly to telephone numbers, IP addresses, E-mail addresses, RFIDs related UID [7] and EPC codes [8], etc. This global uniqueness will cause a location privacy problem.

4. PRIVACY ENHANCED ACTIVE TAG

4.1 System Architecture

In order to resolve the privacy problem, we adopt the basic architecture shown in Figure 1.

At a transmission opportunity, the tag outputs a temporary ID called an Anonymous-ID. This Anonymous-ID transmits a different random value each time. For this reason, if the Anonymous-IDs are collected and analyzed by eavesdroppers, the IDs can only be recognized as unrelated random number sequences, and they cannot be determined to be from the same ID. Certainly, the frequency that the Anonymous-ID is updated can be changed to satisfy the privacy protection level.

The security server decrypts the Anonymous-ID into the original ID. The decoded results are obtained only by a reader that has the acquisition authorization for that ID. In this way, the threats to content privacy and location privacy caused by readers without authorization having unlimited access can be avoided. The reader authentication, its ID acquisition authorization, and secure communications between the reader and the server, take advantage of the existing Internet security technologies.

4.2 Anonymous-ID Generation Methods

We developed the three schemes described below as methods for generating the Anonymous-ID.

[A: Probabilistic encryption scheme]

Inside the tag, a probabilistic public key encryption scheme is implemented, and this scheme generates a different

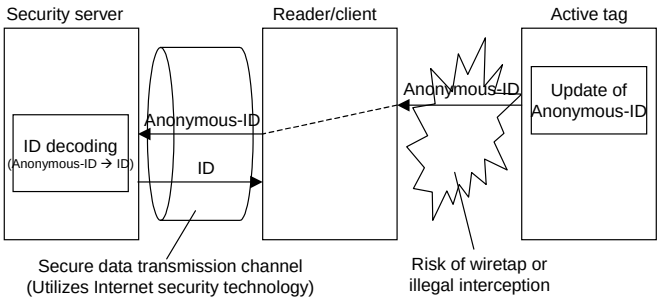


Figure 1. Basic architecture

Anonymous-ID each time. Probabilistic encryption is an encoding scheme in which a different cipher text is generated each time and it is difficult to determine the degree of relatedness among the generated cipher texts. More specifically, even if the same ID is encrypted, the first encryption results and the second encryption results are totally different and unlinkable. Since, in this scheme, information such as the secret key is not stored in the tag it

is highly resistant to tampering. However, since the ID is stored as plain text, it is possible that the ID can be disclosed by tampering. Whether or not this type of self-disclosure can be linked to a threat to privacy depends on the circumstances. For example, as described in Section 3.1, if personal information is stored in the ID, this poses a problem.

In regard to these problems, a function called re-encryption is effective. In this re-encryption function, without decryption one cipher text is generated from another cipher text by using only the public key. Regardless of the number of times re-encryption is performed, the plain text can be obtained by performing decoding once. If this re-encryption function is used, the encrypted ID can be stored inside the tag, and even the danger of disclosing the original ID due to tampering can be abated. For example, the elliptical curve ElGamal is a probabilistic encryption algorithm with such a re-encryption function.

[B: Common key encryption scheme]

When public key encryption, which incurs a large calculation load, is used in the probabilistic encryption scheme, the battery life is curtailed in applications such as the active tag, which has limited calculation resources. To address this, we propose using a method that employs common key encryption, which has a far lower calculation load compared to that for public key encryption. Common key encryption itself does not provide properties such as probabilistic encoding and re-encryption. Common key encryption and random number generation are implemented in the tag, and the original ID and secret key are stored in the tag as well. When the ID is updated, a random number is generated, and then the ID and the random number are combined and encrypted by the secret key. Therefore, each time a different Anonymous-ID can be generated.

In comparison to the probabilistic encryption scheme, the common key encryption scheme has a small calculation load; however, since the secret key must be stored in the tag, it is extremely vulnerable to tampering. Since the secret key must be shared among multiple tags, when disclosing the secret key other tags can also be decrypted and privacy can no longer be protected. The reason that the secret key must be shared is described in the following. If the secret keys are individualized, the server must know which secret key to use for the decoding. In order to make that discrimination, additional information such as an ID key number must be included, and the fixed and unique characteristics of this form would cause new privacy violations.

[C: Hash-chain scheme]

In order to address the issues related to the probabilistic encryption scheme and common key encryption, we believe that applying the Hash-chain scheme [9], which we previously developed for the passive tag, is effective. Hereafter, a simple explanation of the function of the Hash-chain scheme is given using Figure 2. When the tag updates the ID, (1) local variable α is input into Hash function H and (2) α is updated. Next, (3) α is input into Hash function G, and (4) Hash value β is updated as the Anonymous-ID. At the next transmission opportunity, the tag transmits β. The corresponding relationships between the original ID and the initial value of α are safely managed in the server as secret information.

Based on the randomness of Hash function G, the Anonymous-IDs, β, generated each time are different and unlinkable to one another. Since this process is one way, there is no way to retrieve the internal secret information, α, from β. The secret information inside the tag, α, is updated one-way each time α is read using Hash function H. For this reason, even if a third party knows α through tampering, the third party cannot know the retroactive values of α. As a result, previous values of the Anonymous-ID, β, cannot be investigated.

In this way, even if tampering of the secret information in the tag occurs, the previous information up to that point (cipher text, signature, etc.) is protected by the characteristic called forward security. The Hash-chain scheme provides this characteristic.

However, the main issue of this scheme is the limited scalability of resolving the IDs at the security server. Different from encryption, hash functions are one-way functions. For this reason, to resolve the original ID, the server must repeat its calculation until it obtains the identical match to the Anonymous-ID (β) received from the tag by retesting the same procedures that are performed by the tag for each of the initial values of α, which has a one-to-one correspondence to the original ID. As a result, as the number of IDs managed at the server increases the decoding processing time increases. However, if the server disk capacity is sufficiently large that the corresponding tables for all of the β values and IDs can be generated beforehand, the IDs can be resolved in a $\log_2 (N \times M)$ level of retrieval processing time, where N is the number of IDs and M is the envisioned maximum number of reads.

Among the three schemes described above, there are advantages and disadvantages in terms of the calculation load of the tag, safety, and the server load. The results are given in Table 1. We can select the appropriate scheme among the three according to the system requirements.

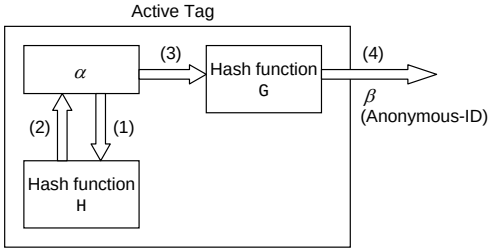


Figure 2. Hash-chain scheme

Table I. Comparison of Privacy Protection Schemes

		A: Probabilistic encryption scheme	B: Common key encryption scheme	C: Hash-chain scheme
Privacy	Content privacy	Protected (o)	Protected (o)	Protected (o)
	Location privacy	Protected (o)	Protected (o)	Protected (o)
	Vulnerability to tamper	Tamper free (o)	Vulnerable (x)	Tamper free (o)
Calculation load	RFID Tag	High (x)	Low (o)	Low (o)
	Security server	Low (o)	Low (o)	High (x)

4.3 Prototype System

In the currently commercialized version of the active tag, some limited functions are provided such as initializing the ID or changing the transmission timing. However, there have been no tags in which a software program for the above-proposed privacy protection schemes can be implemented. For this reason, we developed an active tag that integrates a microprocessor (Figure 3).

The specifications for the developed active tag are given in Table 2. The transmission period and ID update timing are specified so that they are independent of each other. Based on this, requirements such as the level of privacy protection and battery life can be flexibly satisfied. There are two choices for transmission timing, the periodic transmission type and the ultrasound response type in which an ultrasound is received from an outside source and when there is a transmission opportunity a response is transmitted. The ultrasound response type is appropriate for real-time systems such as automatic ticket examination, and

Table 2. Specifications for active tag prototype

General specifications		Wireless specifications		MPU specifications	
Exterior	30x70x15 mm	Transmission frequency	315 MHz±40 KHz	MPU	Motorola 8 bit MPU
Consumed electrical Current (Waiting)	Less than 20 μA	Transmission speed	19.2 kbps	MPU execution frequency	20 MHz
Consumed electrical current (Transmitting)	Approx. 6 mA	Transmission output	500 μV/m@3m	Memory	Flash 60 KB, RAM 4 KB

for event driven systems such as changing the ID transmission interval when entering a store and ID update.

Table 3 presents the implemented Probabilistic encryption and Hash-chain schemes and their respective encryption or hash algorithms and processing time results for updating the ID. From these results, in the Probabilistic encryption scheme it is difficult to update the ID in a short period such as a second, and even if the period is extended to several seconds, since the conditions are such that the processor must constantly be in operation, the battery life becomes extremely short. Since the speed at which people move is limited, an ID update per hour should be more than sufficient. For these requirements, practical application is more than possible. Furthermore, if the encryption processor used in IC cards is employed, calculation at high-speed and with low power consumption is possible in public key encryption.

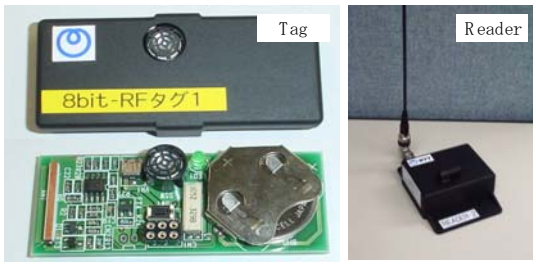


Figure 3. Active tag prototype

Table 3. Processing time

	Encryption / Hash algorithms	Tag processing time
Probabilistic encryption scheme	Elliptic curve ElGamal (key length 160 bits)	6 sec
Hash-chain scheme	SHA-1, MD5	Less than 1 sec

5. CONCLUSION

By using the bearer type active tag, identifying people and providing personal contextual information such as location will become easier and more certain. This type of high-level identification or contextual information collection will become important as a very fundamental technology in the anticipated ubiquitous society. However, a problem is arising where the excellent tracking ability of active tags is abused and personal privacy is being violated.

This paper clarified the active tag privacy problem and proposed three schemes each with different characteristics from the viewpoints of safety, tag calculation cost, and server calculation cost. Furthermore, we constructed an active tag prototype that enables ID encryption and restriction-less update control, implemented two proposed schemes, and evaluated them.

In the future, with the view of achieving a safe ubiquitous society, we plan to intensify the analysis of the application areas for active tags and refine the requirements while investigating a way to achieve the optimal privacy protection scheme.

REFERENCES

1. NIKKEIBP, http://itpro.nikkeibp.co.jp/free/NBY/NEWS/20040630/2/ (in Japanese).
2. RFID journal, “Surveillance System Links Video to RFID Tags”.
3. CNET News.com, “Japan school kids to be tagged with RFID chips”.
4. The Mercury News, “ID Badges on Children”.
5. CASPIAN’s homepage, http://www.spychips.com/
6. Electronic Frontier Foundation, “Mandatory Student ID Cards Contain RFIDs”.
7. ISO/IEC 15963, “Information technology -- Radio frequency identification for item management -- Unique identification for RF tags”, 2004.
8. EPC global, “EPC Tag Data Standards Version 1.1”, 2004.
9. M. Ohkubo, K. Suzuki, S. Kinoshita, “Cryptographic Approach to “Privacy Friendly” Tags,” RFID Privacy Workshop @ MIT, Nov. 2003.

   http://www.rfidprivacy.org.

Conflict Resolution Method utilizing Context History for Context-Aware Applications*

Choonsung Shin and Woontack Woo
GIST U-VR Lab.
Gwangju 500-712, S.Korea
{cshin, wwoo}@gist.ac.kr

ABSTRACT
In this paper, we propose Conflict Manager to resolve conflicts for context-aware applications in smart home environments. Conflicts arise when multiple users access an application or when various applications share limited resources to provide services. In order to resolve conflicts among users, the Conflict Manager assigns priority to each user so that the user with the highest priority can be selected by exploiting conflict history of users. In addition, Conflict Manager detects and resolves conflicts among applications by utilizing preferences of users and properties of the services. To show the usefulness of the proposed conflict resolution method, we apply the proposed conflict resolution method to ubiHome, a smart home test-bed. The experimental results proved that Conflict Manager enable context-aware applications to offer personalized services to multiple users by resolving service conflicts among applications as well as among users.

Keywords
Context-Awareness, service conflict, context history

INTRODUCTION
The aim of ubiquitous computing is to provide users with intelligent services based on the information obtained from distributed but invisible computing resources. These services do not require any cumbersome interface or leaning procedures for users to use them. Especially context-aware applications offer appropriate services to users by utilizing contextual information of environment including users [1]. This information is obtained from various sensors or computing resources distributed in our daily life. However, conflicts occur in context-aware applications when multiple users share the applications or these applications share the limited resources in environment. Service conflict among users is the scenario when multi-users access an application, and then the application have to choose one user to provide a customized service. As a result, the applications could not make a suitable decision to start a service, and each user may not receive personalized services. Resource conflicts also occur among services if each service attempts to share resources at the same time. Consequently, applications start serving to the users without possessing all the necessary resources and thus may result in unsatisfactory services.

Over the last decade, most research, aimed on resolving conflicts, has been done on smart home and intelligent office. Reactive Behavioral System (ReBa) supports conflict resolution among devices in office environment such as, between electric lamps, display devices, and telephones [2]. RCSM (Reconfigurable Context-Sensitive Middleware for Pervasive Computing), an object-based framework, makes sensors and application services independent, forms ad-hoc communication between them, and delivers the necessary context to the applications [3].

However, context management in the previous research has various limitations when they are applied to multi-user environment with various applications. In the case of ReBa, it is difficult to provide to each user with particular services because ReBa focuses on the service for grouped users by inferring main activities from the environment [2]. In RCSM, context management does not consider shared devices or services because contextual information services are provided only through individual device possessed by each user [3].

In this paper, we propose Conflict Manager to resolve service conflict caused by the use of applications among multiple users and limited resources among multiple applications. The proposed Conflict Manager consists of three parts: i) User Conflict Manager which resolves conflict among users , ii) Service Conflict Manager which resolves conflict among services, and iii) Conflict History Manager which assigns priority to conflicting context by utilizing conflict history. Conflict Manager resolves the conflicts among users by choosing a user having the highest priority. In addition, the proposed Conflict Manager detects and resolves conflicts among applications by utilizing properties of services and relationship among them.

* This works was supported by DSC of Samsung Electronics Co., Ltd., in Korea

This paper is composed as follows. First of all, we introduce service conflicts caused by multiple users and multiple applications in context-aware computing environments. We also classify the service conflicts into three types according to conflict sources. We then describe Unified Context-aware Application Model for ubiquitous computing environment (ubi-UCAM). We then introduce Conflict Manager which resolves services conflicts among users and among applications. Finally, we explain the experimental results of applying this method to ubiHome test-bed.

CONFLICTS IN CONTEXT-AWARE APPLLICATIONS

In context-aware computing environments, various applications provide users with customized services based on users' contexts within a service area. In order to provide the services, the applications require one or more resources, such as display device, sound device, light device, or, etc, according to their properties. Furthermore, in such service environments, the number of users accessing the same applications is not limited.

Unlike single user and single service environment, applications in the computing environment have to respond while considering other applications and various users within a services area. We define such situation as a service conflict. We classify the conflict into three types according to sources of conflicts: service conflicts among multiple users, service conflicts among multiple applications and service conflicts among multiple users and multiple applications. Service conflicts among users are caused due to use of an application by multiple users. In this situation, the application has to choose one customized service. For example, a service conflict arises when users A and B are trying to watch their preferred broadcasts from television service. Service conflicts among multiple applications are caused by providing of services among multiple applications. Due to the conflict, the application cannot provide users with customized responses. For instance, this kind of conflicts occurs when television application and music application start to provide their customized services simultaneously. Service conflicts among users and applications are caused due to the use of multiple services by multiple users. This kind of conflict is similar to the conflict among applications, but the users assigned to the applications are different. For example, a service conflict arises when user A is trying to use a television application while user B is trying to use a music application.

To deal with these conflicts, resolution methods have to resolve the conflict according to sources of conflicts. Furthermore, in order to reflect the change of users' preferences and their environment, the conflict resolution methods must adapt to users and their environment. In this paper, we deal with two kinds of conflicts, i.e. among users and among services, by utilizing conflict history of users as well as user contexts and service profiles.

UNIFIED CONTEXT-AWARE APPLICATION MODEL

In order to deal with service conflicts, we adopt Unified Context aware-Application Model for ubiquitous computing environment (ubi-UCAM) [4]. The ubi-UCAM is a context-based application model to provide users with personalized services by exploiting context in ubiquitous computing environments. In addition, to ensure independence between sensors and services, the ubi-UCAM utilizes unified context represented as 5W1H (Who, What, Where, When, How and Why) [4]. The ubi-UCAM employs different types of unified context based on the role of each context. These include preliminary context, integrated context, conditional context, and final context. Figure 1 shows the overall architecture of the ubi-UCAM.

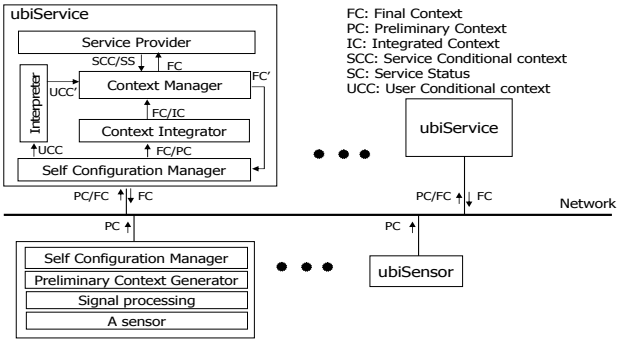


Figure 1. ubi-UCAM

As shown in Figure 1, the ubi-UCAM is composed of ubiSensors, a sensor, and ubiServices, an application to provide a service. Each ubiSensor generates a preliminary context from the features extracted from a physical sensor. It then delivers the preliminary context to ubiServices within a service area. Each ubiService collects preliminary contexts as well as final contexts delivered from other ubiServices within a service area. The ubiService then builds integrated context of each user by classifying the preliminary contexts and final contexts. It searches conditional context from a Hash table, which manages specific service action and condition, corresponding to each integrated context. It generates a final context to be used by Service Provider after resolving conflicts among users and services. Finally, ubiService executes appropriate action with parameters described in the final context. It utilizes application-specified methods which are programmed by application developers.

CONFLIICT MANAGEMENT

In ubi-UCAM, service conflicts occur not only due to multiple users who access ubiServices at the same time, but also due to multiple ubiServices trying to share resources in their surrounding. To resolve service conflicts among users, the proposed Conflict Manager assigns priority to users and chooses the user given the highest priority. In addition, to deal with service conflict among ubiServices, the Conflict Manager detects and resolves conflicts, based on the properties of ubiServices and relationship between

them. Moreover, priority of users and ubiServices are not fixed, but adapts to user's preference and behaviors. Therefore, the Conflict Manager not only resolves conflicts among users and among ubiServices, but also dynamically assigns priority to users and ubiServices.

Conflict Resolution among Users

User Conflict Manager resolves conflicts caused by users who try to use ubiServices within a service area. To resolve the conflict, User Conflict Manager manipulates user contexts in two steps: building a conflict list and selecting a proper user from it. Figure 2 depicts the overall procedure of User Conflict Manager.

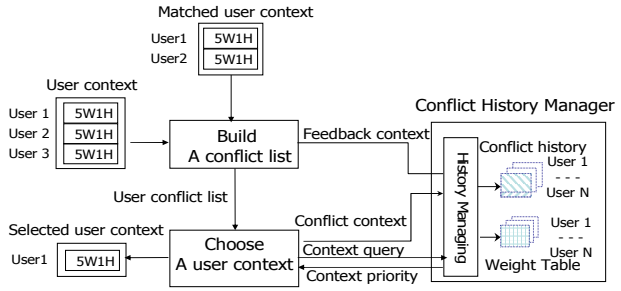


Figure 2. User Conflict Manager

As shown in Figure 2, User Conflict Manager makes a conflict list of matched user context on users who are expected to cause conflict among users, including those who are currently using the service. In this process, users who leave the service area are excluded from the list because we assume they do not want to use the service any more. In addition, user's feedback is also delivered to Conflict History Manager. The context is considered as user feedback if there is user implicit context such as remote controller. In the next stage, User Conflict Manager chooses one user from the conflict list based on user's priority calculated from Conflict History Manager according to user context. In this process, conflicts are handled in several ways according to the number of users within the service area. In the case of one user situation, we know that there is no conflict among users. Therefore, User Conflict Manager just selects the user context as a result of conflict resolution. However, we have to consider the situation when there are more than two users in a service area. In this situation, User Conflict Manager selects the user having highest priority because conflicts may occur. In addition, it notifies the result of conflict resolution to enable Conflict History Manager to store conflict context.

Conflict Resolution among ubiServices

Service Conflict Manager resolves services conflicts caused by multiple ubiServices trying to share resource in the service area. The conflicts are caused by not only a ubiService itself but also other ubiServices. Therefore, Service Conflict Manager deals with the conflict in two ways: conflict caused by other ubiServices and conflict caused by a ubiService itself. Figure 3 shows Service Conflict Manager.

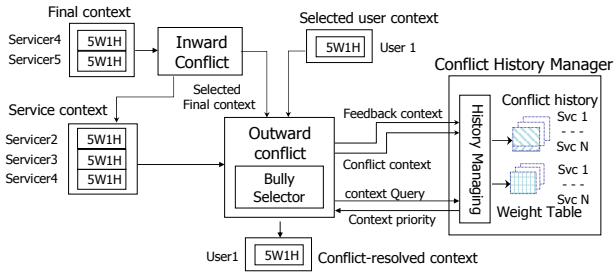


Figure 3. Service Conflict Manager

As shown in Figure 3, Service Conflict Manager creates a context which contains information about the ubiService and its stop action, if resources involved in other ubiServices are the same as those of the ubiService itself. As a result, the application responds to changes of other ubiServices which cause conflict, using final contexts coming from other services. In addition, Service Conflict Manager updates the final context to the final context table. Service Conflict Manager prevents this ubiService causing conflict with other ubiServices. To detect possible conflicts, it checks to see if there are any services using the same resource before delivering the context. Service Conflict Manager compares priority of the service contexts calculated from Conflict History Manager if there are conflict services within a service area. Finally, it sends the conflict-resolved context to Final Context Generator when there aren't any services related to the same resource. In addition, Service Conflict Manager just sends the resolved context to Conflict History Manager to notify the result of conflict resolution.

Service Conflict Manager also deals with the situation when multiple services want to use resources at the same time. This is because ubiServices can respond to the same condition. In the case of this conflict, several ubiServices want to use the same resource. For example, television and movie services can be triggered at the same time when a user enters home. To deal with this situation, we adopt bully algorithm that elects a leader among processes in distributed computing environment. The algorithm chooses a coordinator having the highest priority among processes [7]. In service conflict, the algorithm is used to choose the highest ubiService among ubiServices which try to use shared resources.

Conflict History Management

Conflict History Manager takes charge of maintaining conflict history and determining priority of conflicting context. To efficiently use the limited storage, it only maintains conflict history for a short period of time. In addition, to reflect user preference, Conflict History Manager calculates the priority of conflicting contexts based on Bayes theory which is widely used for

classification or prediction. Figure 4 shows the overall architecture of Conflict History Manager.

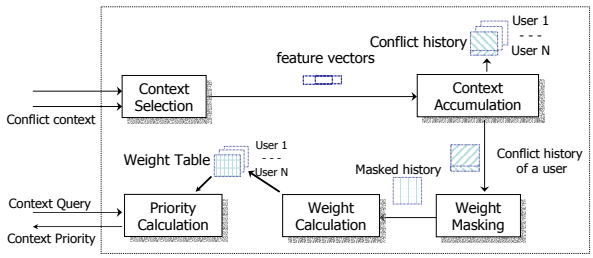


Figure 4. Conflict History Manager

As shown in Figure 4, Conflict History Manager receives feedback and conflicting contexts of users from Conflict Manager. Based on the contexts, Conflict History Manager generates a feature vector containing information about the conflict situation. Afterwards, the feature vector is stored in a history file so that it can be retrieved whenever required. Conflict History Manager then loads the feature vectors, related to a specific user, from conflict history. Conflict History Manager recalculates weights of conflicting contexts based on the feature vectors. In order to obtain the weight, Conflict History Manager applies Bayesian theory to the feature vectors. Equation (1) shows Bayesian theory. In the equation, feature vector X is composed of $(x_1, x_2, x_3, x_4, x_5, x_6)$. Each element of X is mapped to the value of Service type, Location, Time, Gesture, Stress, and Conflicting user. The result of conflict resolution H_j, which is represented by (H_1, H_2) indicates the Target class. Consequently, we obtain probability $P(H_1|X)$, for allowing the current user of a service to continue using the service when conflict arises, by multiplying posteriori probability $(X|H_1)$ and prior probability $P(H_1)$.

$$P(H_j \mid X) = \frac{P(X \mid H_j)P(H_j)}{P(X)} \qquad (1)$$

According to the equation, we assume that a current user of a ubiService will continue using the service in case of a conflict when posteriori priority $P(H_1|X)$ is greater then $P(H_2|X)$. Otherwise, another user uses the service. So, a priority of context is the difference between maximized posteriori probability of $P(X|H_1)(H_1)$ and $P(X|H_2)(H_2)$. Therefore, a weight of each feature is expressed by priori probability of the feature $P(x_k|H_j)=s_{kj}/s_j$. s_{kj} is the number of conflicting contexts having a specific value of s_k within the class H_j class. s_j is the sum of values of conflicting contexts belonging to H_j. Conflict History Manager calculates weights of conflicting contexts of users based on the weight table. The calculated results are updated in hash-table and a weight file for future search.

Conflict History Manager provides priority of the conflicting context based on the weight table when Conflict Manager requests priority for a conflicting context Conflict History Manager retrieves weights of the user, identified by 'Who' context of conflicting context, from the hash-table. Afterwards, it applies the weights to the conflicting context to Equation (2) to calculate posteriori probability. The Conflict History Manager calculates posteriori probability $P(X_i|H_1)$ when a current user will continue using the service, and posteriori probability as $P(X_i|H_2)$ when another user will use it.

$$P(X_i \mid H_j) = \prod_{k=1}^{n} P(x_k \mid H_j) \qquad (2)$$

Finally, Conflict History Manager calculates a priority of the conflicting context. Equation (3) shows the priority of conflicting context. In the equation, $P(X|H_1)P(H_1)$ is the maximized probability of the current user to continue using the service. $P(X|H_2)P(H_2)$ is the maximized probability of another user to use the service. Conflict History Manager delivers the difference of these two probabilities to Conflict Manager as a priority of the conflicting context.

$$Priority\ (X_i) = P(X_i \mid H_1)P(H_1) - P(X_i|H_2)P(H_2) \qquad (3)$$

Based on the conflict history and Bayesian theory, Conflict History Manager adjusts the weight of conflicting context using conflict history of users after conflicts are resolved. It also assigns a priority to conflicting contexts of users based on the weight table when conflicts arise.

IMPLEMENTATION AND EXPERIMENT

We have evaluated the effectiveness of the conflict resolution method based on the ubiHome test-bed. The proposed Conflict Manager selects one among several users when multiple users attempt to access their registered service. In addition, it decides to provide the service when priority of the service is higher than the other services located within a service area. Finally, we also measured accuracy of the proposed method with four family members

Experimental Setup

The proposed Conflict Manager was implemented with J2SDK 1.4™ so that it can be applied to various applications. As shown in Figure 5, we tested Conflict Manager in ubiHome, a smart home test-bed at GIST [5].

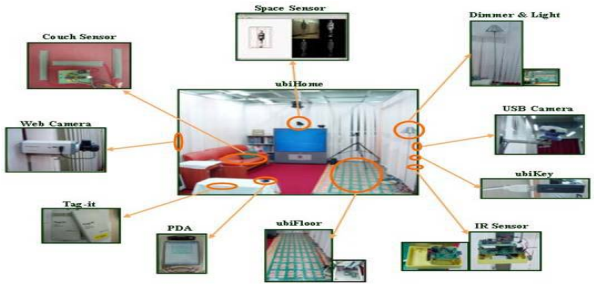


Figure 5. ubiHome test-bed

As shown in Figure 5, we utilized various ubiServices such as, television service, Internet service, music service, movie service, light services, etc. These ubiServices offer customized services to users. In addition to the services, we also exploited various sensors: ubiCouch sensors,

ubiTrack, and ubiRemocon. The ubiCouch sensors, comprised of on/off switches, detect user's behaviors. The ubiTrack is infrared-based location tracking system that tracks users' location [6]. The ubiRemocons are a kind of remote controllers, implemented with Personal Java, to control these services.

Experimental Analysis

In order to measure accuracy of resolution method of the proposed Conflict Manager, we experimented on user conflict in two ways: i) a resolution method based on the Bayesian theory and, ii) a resolution methods having fixed priority. To test two methods, we employed television service that users use in a home environment. While using the television service, family members cause conflicts due to their preferences and its broadcasts. In our experiment, the television service selects a preferred program a user. It decided a specific program of the user who has the highest priority according to each selection strategy when conflicts occurred. The service gathered feedback from users in pre-defined amount of time and judged the accuracy on the selection. The television service counts the number of "incorrectness" and "correctness" of the selection. As the result of the selection, we have built confusion matrix to know how well it works. Table 1 shows the experimental results of the proposed conflict resolution method

Table 1. Confusion matrix for conflict resolution (unit: %)

Users	Father	Mother	Son	Daughter
Father	81	8	4	7
Mother	8	79	7	6
Son	4	3	78	15
Daughter	5	6	14	75

As shown in table 1, the proposed resolution method provides the television service to other users who have lower priority in the conflict resolution having fixed priority. This is because the conflict resolution method assigned priorities to users based on their context. In addition, accuracy of the resolution method was relatively higher than the resolution method having fixed priority. The improvement of accuracy was due to the fact the resolution method reflected the changes of their preference and resolution policy. Therefore, conflict solution could resolve service conflicts caused due to use of services by multiple users.

In addition, we configured properties of services to deal with conflict among services. In our experimental setup, all the services were in the same area. Especially, television, and movie services were operated on the same computer. Based on the properties, we monitored the services in ubiHome in order to observe resource conflicts among services. Table 2 shows the number of conflict among services.

Table 2. The number of conflict among services (unit: %)

Services	Television	Movie	Music	Light
Television	-	33	56	11
Movie	54	-	25	21
Music	72	28	-	-
Light	77	23	-	-

In case of television service, most of the conflicts are related to movie service. The rest of the conflicts are associated with movie and music service. Movie service, which shares sound, light, and display resource, is related to all the services. In particular, conflicts of movie service are mostly due to television service which is accessed by users frequently. Besides, movie service also conflicts with light service since the services use light resource. Music service was related to television and movie service using sound and display resources.

CONCLUSION

In this paper, we proposed the Conflict Manager to resolve services conflicts among users and among applications. In order to resolve conflicts among users, the proposed Conflict Manager maintained the conflict history of users, calculated the priority of user context with Bayes theory, and then selected one user. In addition, Conflict Manager detected conflicts among applications based on properties of each service. These conflicts were resolved with the priority so that the applications provided services without causing conflicts. In our future works, however, we will employ additional applications deal with the conflicts. We will also observe the conflicts with users' behaviors over longer periods.

REFERENCES

1. Anind K. Dey, "Understanding and Using Context. Personal and Ubiquitous Computing, Special issue on Situated Interaction and Ubiquitous Computing, 5(1),. (2001)
2. Nicholas Hanssens, Ajay Kulkarni, Rattapoom Tuchinda, and Tyler Horton, "Building Agent-Based Intelligent Workspaces," In *ABA Conference Proceedings*, June. (2002)
3. S. S. Yau, F. Karim, Y. Wang, B. Wang, and S.Gupta, "Reconfigurable Context-Sensitive Middleware for Pervasive Computing," *IEEE Pervasive Computing, joint special issue with IEEE Personal Communications*, 1(3), pp.33-40, July-September. (2002)
4. S.Jang, and W.Woo, "ubi-UCAM: A Unified Context-Aware Application Model", Lecture Note Artificial Intelligence (*Context'03*), Vol, 2680, pp.178-189, 2003
5. Y.Oh, W.Woo, "A unified Application Service Model for ubiHome by Exploiting Intelligent Context-Awareness," *Proc. Of Second Intern. Symp. On Ubiquitous Computing systems (UCS2004),* pp. 117-122,2004.
6. S.Jung, W.Woo, " UbiTrack: Infrared-based user Tracking System for indoor environment," *ICAT'04*, 1, paper 1, pp. 181-184. (2004)
7. Garcia-Molina, H. Elections in Distributed Computer Systems. *IEEE Transactions on Computers*, Vol, C-31, No. 1, pp. 48-59. (1982)

Choonsung shin received the B.S degree in Computer Science from Soongsil University in 2004. Now he is a M.S. student in Department of Information and Communications, Gwangju Institute of Science and Technology (GIST) since 2004.

Research Interest: Context Awareness, Human Computer Interaction, Ubiquitous/ Wearable Computing.

Woontack Woo received his B.S. degree in EE from Kyungpook National University, Daegu, Korea, in 1989 and M.S. degree in EE from POSTECH, Pohang, Korea, in 1991. He received his Ph. D. in EE-Systems from University of Southern California, Los Angeles, USA. During 1999-2001, as an invited researcher, he worked for ATR, Kyoto, Japan. In 2001, as an Assistant Professor, he joined Gwangju Institute of Science and Technology (GIST), Gwangju, Korea and now at GIST he is leading U-VR Lab.

Research Interest: 3D computer vision and its applications including attentive AR and mediated reality, HCI, affective sensing and context-aware for ubiquitous computing, etc.