Research Abstract for ICSE 2003 Doctoral Symposium

Self-Extending Systems for Context-Aware Mobile Computing

Asa MacWilliams, *macwilli@in.tum.de*
Advisor: Bernd Brügge, *bruegge@in.tum.de*
Institut für Informatik, Technische Universität München
Boltzmannstrasse 3, 85748 Garching bei München, Germany
Phone +49-89-289-18228, Fax +49-89-289-18207

SUMMARY: *Self-extending systems* are well suited to context-aware mobile computing. They adapt to users' desires and allow developers to add functionality at run time.

RESEARCH AREA: Development process for mobile systems

KEY WORDS: software development process, component-based systems, middleware, context-aware computing, augmented reality

1 Goals

In my thesis, I wish to address the problem of how to develop software for context-aware mobile computing that can adapt to constantly changing requirements at run time.

By *context-aware mobile computing*, I understand a mode of computing characterized by three features. First, the user is involved with tasks in the *real world* such as navigation, car repair, building design or control of complex systems. Second, the system *assists the user* with these tasks by providing appropriate information, facilitating collaboration and controlling machines. And third, the system is *ubiquitous*, i.e. it combines stationary resources such as databases, cameras for position tracking and projection screens with mobile resources such as handhelds, wearable computers and head-mounted displays.

Examples include a pedestrian campus navigation system (Figure 1a, [2]), a wearable computer assisting a mechanic repairing a broken-down car [13], a system for architectural design and visualization on a building site [1] or a collaborative multimodal game (Figure 1b, [12]). Context-aware mobile computing uses techniques from wearable computing, pervasive computing, and augmented reality.

Compared to "classical" desktop software, developing software for these systems is difficult. Requirements are hard to elicit from blue-collar end users, and at run time, the users move around in a changing environment, rather than sitting in front of a desktop computer.

In my thesis, I propose a software development method that address these challenges. It enables users, developers, and the system itself to adapt the system to changing requirements at run time.

2 Related Work

In my master's thesis [7], I showed that *self-organizing systems* provide a viable method to build mobile augmented reality systems. DWARF, the Distributed Wearable Augmented Reality Framework [2, 3, 5], is designed as a collection of software services that can run on separate hardware components connected by wired or wireless networks. These components can be distributed on the body, e.g. on a belt, or as external devices in intelligent environments. The services discover each other and dynamically cooperate to form a complete system.

In building several different example systems [2, 12, 13], I have experienced the inadequacy of classical software development processes and development tools. Especially, I missed visualization and monitoring tools for distributed, multi-platform systems, requirements elicitation and analysis methods for blue-collar applications and methods to deal with changing system design and deployment.

Several other research groups are building context-aware mobile computing systems [6, 9, 10, 14], and software libraries are becoming available for augmented reality [4] and context-aware computing [11]. However, while some of these projects focus on software architecture [8], none address the software development process.

3 Hypothesis

My research hypothesis is that *self-extending systems* are well suited to context-aware mobile computing. These systems have three properties: Their architecture is based on distributed components; their middleware allows them to self-organize according to the users' desires; and their development process allows developers to add missing functionality at run time. Such systems allow useful behavior to emerge that no single developer or user anticipated.

4 Proposed Solution

In my thesis, I propose a new development method allowing incremental requirements analysis and system design at system run time. This turns *self-organizing* systems into *self-extending* systems. Such a system is characterized by its architecture, middleware, and development process.

Architecture The system consists of software *services* on mobile and stationary computers. Each service has a well-defined functionality that is understandable in the same fashion to both the user and the developer. Basic services use low-level hardware to gather contextual information and interact with the user; e.g. a speech recognition service, a three-dimensional rendering service or a hand position tracking service. Higher-level services aggregate the context information and enable multi-modal interaction; e.g. a petri net for coordinating multimodal input or a prediction service for tracking data. The highest-level services are applications: tasks or roles the user can identify, such as "I want to repair this car."

This is the software architecture currently used in DWARF [2].

Middleware A service is realized by a separate operating system process or thread. The services communicate using various interprocess or network communication methods and are connected together dynamically using a middleware component, the *service manager*, which runs on each host in the network.

Each service has a description of its *abilities* (the functionality it provides to the user or to other services), and its *needs* (functionality of other services it depends on). A low-level service could have the ability "provide position information for optical marker X"; a high-level service could have the ability to "provide repair instructions for car X".

The middleware uses these service descriptions to start new services and to connect services together, building a directed graph of communicating services. The service descriptions are parametrizable, so that the middleware can change services' behavior at run time based on the users' desires and on the availability of other services.

There is no central component that coordinates all services; rather, distributed middleware decomposes the users' intent and propagates it through the system, mapping it onto ever smaller services. The system can thus handle mobility and network disconnectivity more easily than a centralized system, but at the same time offer coherent and useful behavior to the user.

I built a prototype of this middleware, based on CORBA, in my master's thesis [7].

Process I propose an entity-oriented software development process based on services. This uses a single model for requirements analysis entities (user tasks, use cases), system design entities (services, needs, abilities), and implementation entities (processes, event channels).

The system includes high-level services that allow the user to provide feedback and request additional features. These wishes are linked directly to the requirements analysis model and the descriptions of running services, so developers can easily program new services to provide the missing functionality. Monitoring and debugging tools let the developer test the new services in the running system.

By identifying and adding missing functionality, users and developers cooperate to incrementally improve a continuously running system.

(a) Augmented reality navigation; view through head-mounted display

(b) Pervasive computing with embedded system; player scoops virtual sheep off projector table with palmtop

(c) Monitoring tool for distributed services; system configuration can be changed at run time

Figure 1: Mobile context-aware computing applications and development tool

5 Expected Contribution

In my thesis, I will provide the description of my proposed development process, along with supporting middleware and development tools, e.g. for testing, monitoring, visualization, documentation and user feedback.

I have already made progress in the three sub-areas of my work: architecture (a comparison of software architectures for augmented reality [8]), middleware (CORBA on embedded and wearable devices, Figure 1b), and process (a monitoring and visualization tool for distributed, dynamically communicating services, Figure 1c).

6 Methods

My Ph.D. work started in October 2001, and I intend to be finished by October 2004. I will validate my concepts using an experimental system approach within the DWARF project. Here, I can leverage the services other members of the group are developing, and supervise groups of students in developing example systems.

In group projects with between 5 and 50 senior-level students, we have already built several systems – for campus navigation (Figure 1a); prototype vehicle presentation; automobile repair; robot maintenance; and multimodal, multiuser entertainment (Figure 1b). Other projects, e.g. for architectural design and visualization, are currently underway, and each will improve on the architecture, middleware and process, and serve as a testbed for evaluation and validation of the concepts of self-extending systems.

References

[1] Archie – augmented reality collaborative home improvement environment. Technische Universität München, http://wwwbruegge.in.tum.de/projects/DWARF/projects/archie.

[2] Martin Bauer, Bernd Bruegge, Gudrun Klinker, Asa MacWilliams, Thomas Reicher, Stefan Riss, Christian Sandor, and Martin Wagner. Design of a component–based augmented reality framework. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality – ISAR 2001*, New York, USA, 2001.

[3] Martin Bauer, Bernd Bruegge, Gudrun Klinker, Asa MacWilliams, Thomas Reicher, Christian Sandor, and Martin Wagner. An architecture concept for ubiquitous computing aware wearable computers. In *Proceedings of the 2nd International Workshop on Smart Appliances and Wearable Computing (IWSAWC 2002)*, Vienna, Austria, July 2002.

[4] Marc Billinghurst, Hirokazu Kato, and Ivan Poupyrev. The magicbook - moving seamlessly between reality and virtuality. *IEEE Computer Graphics and Applications*, Vol.21(No.3), 2001.

[5] Dwarf – distributed wearable augmented reality framework. Technische Universität München, http://wwwbruegge.in.tum.de/projects/DWARF.

[6] R. L. Lagendijk. Ubiquitous communications (ubicom) - updated technical annex 2000. Technical report, Ubiquitous Communications Program TU-Delft, Jan. 2000.

[7] Asa MacWilliams. Using ad-hoc services for mobile augmented reality systems. Master's thesis, Technische Universität München, Department of Computer Science, Feb. 2001.

[8] Asa MacWilliams and Thomas Reicher. Software architectures for augmented reality systems – report to the ARVIKA consortium. Technical report, Technische Universität München, 2002.

[9] Joseph Newman, David Ingram, and Andy Hopper. Augmented reality in a wide area sentient environment. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality – ISAR 2001*, New York, USA, 2001.

[10] Gerhard Reithmayr and Dieter Schmalstieg. OpenTracker – an open software architecture for reconfigurable tracking based on XML. Technical report, TU Wien, 2000.

[11] Daniel Salber, Anind K. Dey, and Gregory D. Abowd. The context toolkit: Aiding the development of context-enabled applications. In *CHI*, pages 434–441, 1999.

[12] Christian Sandor, Asa MacWilliams, Martin Wagner, Martin Bauer, and Gudrun Klinker. Sheep – the shared environment entertainment pasture. In *IEEE and ACM International Symposium on Mixed and Augmented Reality – ISMAR 2002*, 2002.

[13] Tramp – traveling repair and maintenance platform. Technische Universität München, http://wwwbruegge.in.tum.de/projects/DWARF/projects/tramp.

[14] Z. Wang and D. Garlan. Task-driven computing. Technical report, Carnegie Mellon University School of Computer Science, 2000.