SYSTEM SUPPORT FOR UBIQUITOUS COMPUTING

Erwin Aitenbichler *

Abstract
An important issue in a ubiquitous computing world is that of future personal devices and how they communicate with their environment. This paper discusses smart digital identities that represent user's in the digital world and are able to perform operations such as authenticating the user to other parties. Furthermore, the environment with its many different devices, operating systems and network technologies, poses new challenges to communication middleware.

1. Introduction

Internet is currently enjoying great success as a means for communication between people and between people and computers. Popular Internet services include email, the Web, and more recently mobile services, such as WAP. Advances in mobile networking, as demonstrated by GPRS and UMTS/IMT2000, will bring a greater need of communications. However, the biggest foreseeable growth of communication needs does not come from people-to-people or people-to-computer communications, but from computer-to-computer communications. Smart appliances and future pervasive computing architectures will mean billions of devices, all with communication capabilities and needs. In the future, a large amount of traffic on the Internet will come from devices talking to other devices.

The Mundo project at the Telecooperation Department at Darmstadt University of Technology is concerned with future pervasive and ubiquitous computing infrastructures. A short introduction into the components of Mundo is given in the next section. My work focuses (i) on smart digital identities and (ii) enabling communications between the different entities in Mundo by finding suitable communication abstractions and providing adequate middleware support for application programmers.

This paper is organized as follows. Section 2. gives an overview of Mundo. Section 3. presents the concept of Smart Digital Identities. In Section 4. the properties of a communications infrastructure suitable for ubiquitous computing are discussed. Section 5. concludes the paper.

2. Mundo

This section provides a very brief overview of the Mundo project. A more complete description can be found in [6]. Figure 1 shows the different entities in Mundo that are explained in the following.

*Telecooperation Group, Department of Computer Science, Darmstadt University of Technology, Germany, erwin@tk.informatik.tu-darmstadt.de, http://www.ubicomp.tk/

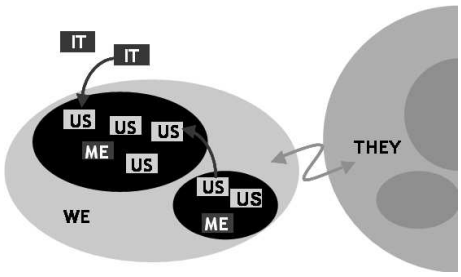

Figure 1. Mundo vertical architecture

ME (Minimal Entity)

We consider ME devices as the representation of their users in the digital world. The personal ME is the only entity always involved in the user's activities. Our decision for one individually owned device rather than a publicly available infrastructure comes from *trust establishment*. As an increasing amount of computer based activities have (potential) legal impact and involve privacy issues, it is vital that users can trust the device carrying out such transactions.

US (Ubiquitous aSsociable object)

Minimalization pressure will not permit feature-rich MEs in the near future. Hence, they must be able to connect to local devices such as memory, processors, displays, and network devices in an ad hoc manner. We call this process *association* and we call such devices *ubiquitous associable objects* (US). Through association, the ME can personalize the US to suit the user's preferences and needs. For privacy reasons, this personalization becomes unavailable if the US gets out of range of the ME.

IT (smart ITem)

There are also numerous smart items that do not support association that would turn them into an US. Vending machines, goods equipped with radio frequency IDs, and landmarks with "what is" functionality are just a few examples. We define such *smart items* as ITs. An IT is any digital or real entity that has an identity and can communicate with the ME. Communication may be active or passive; memory and computation capabilities are optional.

WE (Wireless group Environment)

We expect ad hoc networking to be restricted to an area near to the user of a ME device, as connections with remote services will involve a non-ad hoc network infrastructure. The functionality of a WE is to bring together two or more personal environments consisting of a ME and arbitrary US entities each. A WE makes connections between the devices possible and also allows for sharing and transferring hardware (e.g., US devices) and software or data between WE users.

THEY (Telecooperative Hierarchical ovErlaY)

We regard overlay cells as the backbone infrastructure of Mundo. These *telecooperative hierarchical overlay cells* (THEY) connect users to the (non-local) world, deliver services, and data to the user. THEY support transparent data access, for example, frequently used data may be cached on US devices. They offer cooperation between different physical networks, transparent to the users.

3. The ME device as Smart Digital Identity

The ME device carries the user's digital identity and is able to perform operations such as authenticating the user to other parties. We believe that interaction with the ubiquitous infrastructure will mainly happen through a personal device, owned by each user. This is because it will represent the user in the digital world and carry out transactions, sometimes with only implicit consent of the user. Hence, it is vital that a user is able to *trust* the device carrying out these actions. The easiest way to instill this trust in users is to have them actually *own* the device instead of relying on a networked service, which may remain an abstract concept for many users.

We have decided to base our architecture around a single, minimal device, because such a device is easy to carry around in all situations. Our goal in developing Mundo around a Minimal Entity, is to identify the fundamental requirements. In other words, what functionality must be available to the user at all times? As we present below, any ME can augment its capabilities through association with other entities. However, the user is not required to carry around many devices with her.

Second, the ME is an interface device for interacting with the computers and services of the world. The device can offer basic services in all situations, and offer enhanced services when connected to a network. In terms of pure functionality, i.e., holding the user's digital identity, such a ME device could potentially be constructed in a very simple manner, for example, based on an RFID chip or similar [11, 5]. We have decided to augment this basic functionality by adding audio input/output capabilities, local processing and (limited) storage. Although this makes the device larger, we believe that the demand for audio communications is ubiquitous, as demonstrated by the popularity of mobile phones. Therefore, it is likely that users will (almost) always carry an audio device with them anyway. Furthermore, the TA can act as a trusted terminal that is required to perform secure transactions.

I have been working on actual hardware prototypes of ME devices in the Talking Assistant [1] project. The Talking Assistant (TA) is a voice-centric user terminal in the form factor of a headset. It supports audio in-/output and features a hardware MP3-decoder for enhanced playback quality and more efficient use of resources like memory and network bandwidth. The headset connects wirelessly to the network infrastructure via a Bluetooth radio. A combination of sensors is used to gather context information, such as the location of the mobile user in space and the head orientation in all three rotational axes. The location is determined by an optical tracking system. The heading is determined with an electronic compass and an acceleration sensor is used to measure the tilt angles.

4. Communications

Even though users and their activities are the focus of Mundo, through their MEs, the majority of the communications in Mundo will happen between the different devices. Examples of this are an US associating automatically with a ME, a WE forming for sharing information, an US communicating with another US, and so on. It is therefore vital to have a strong basic communication layer which allows to connect all the components of Mundo in an efficient manner.

From a software engineering point of view, abstracting all the communications of Mundo into the middleware, which is called MundoCore in the following, provides several advantages. First, it allows to separate the communication layer and services. In effect, using MundoCore pulls out all communication related code from service code, making it easier to write new services, since now the service code does not need to handle as many exceptions, deal with broken connections, etc.

4.1. Requirements

A fundamental way in which nomadic computing differs from desktop computing is the great variability of network connectivity. Current applications treat changes in bandwidth or latency as exceptions or errors, but these conditions must be treated as the normal case in nomadic environments. Network access takes place from different locations with a number of different devices by means of different network providers. From past experiences in designing and implementing parts of Mundo, the following key requirements for a pervasive computing communication middleware have been identified:

Modularity. The middleware must have a minimal kernel with the ability to plug in services on-demand, according to the current needs. Pluggable services also offer the possibility for determining the *service fidelity* on-demand. By service fidelity, we mean total quality of the service obtained by combining several pluggable service modules.

Small footprint. Many pervasive computing devices (e.g., the Talking Assistant [1]) have only very limited resources in terms of memory and processing power, hence it is vital that the middleware has a small memory footprint. Being able to plug in services on-demand helps us achieving this goal.

Handovers. The middleware must support both horizontal and vertical handovers. Some devices are likely to be mobile and they will need efficient handovers. Currently, a handover often results in a broken TCP connection which most applications cannot handle well.

Offline support. Even in future communication networks it is unlikely that internet connectivity will be available anywhere and anytime. Furthermore, a user might prefer to stay offline until a high bandwidth connection becomes available to send mass data into the network, because this is cheaper and saves battery lifetime.

Spontaneous Networking. The middleware should be able to automatically discover other nodes in range and perform automatic configuration of network interfaces. If no infrastructure is present, nodes should communicate in a peer-to-peer manner.

Multicast. Because wireless networks are based on a shared medium, multicast would often be ”for free”. However, network technologies like Bluetooth hide this functionality and only provide point-to-point links. Many middleware platforms do not have proper abstractions for point-to-multipoint communication patterns. Multicast functionality should be available up to the application level, to be able to make efficient use of the usually scarce wireless bandwidth (e.g. in streaming applications).

4.2. Concepts

In MundoCore, the single programming abstraction on the basic level is that of a Publish/Subscribe system. Even remote method invocations build on top of this concept. It has been shown in many earlier projects [7] that event routing systems scale well to very large networks. In the following, I use the terminology introduced in [4] for Publish/Subscribe systems.

4.3. Services

Services are components that implement a certain functionality by processing messages from channels or their methods are invoked by means of remote method calls. Services usually realize their

functionality by delegating subtasks to other services. In most cases, the Publish/Subscribe metaphor provides good decoupling between them.

A service is the **migration unit** and manages objects that are closely related to each other. References to local and remote objects are explicitly distinguished by different types of reference variables. Services are not bound to a particular physical location, they can be migrated from one address space to another. The communication abstractions provided by *MundoCore* enable services to maintain all communication links to other services even if they are moved to other physical locations.

The service **manages execution**. By means of sessions, incoming remote method calls are serialized to control concurrency. A session is a single-threaded context for producing and consuming messages, like in JMS (`http://java.sun.com/products/jms`). The service can also be thought of as a layer between the runtime environment and the objects that controls execution. The last two aspects are inspired from the virtual processor concept described in [3].

Multiple services can offer the same interface and realize the same functionality, but rely on services in the network to different grades. We call this concept **service fidelity**. On devices with very limited processing power, simple services run on the device that mainly delegate all tasks to more powerful services in the network. If no network infrastructure is present, the quality of a service may get limited to a basic set of functions. For example, the voice recognition system running on the ME may be limited to understanding a dozen of core commands; once the device has a network connection more complicated commands are serviced by a network-hosted full fledged voice recognition system and the device may also be used for dictating letters, mails, etc.

4.4. Extensions to Publish/Subscribe

A Publish/Subscribe system offers a clear advantage for point-to-multipoint communication patterns. Because we also use this abstraction for point-to-point connections, some optimizations are necessary. Such specific communication patterns are expressed in attributes of the channel objects. The messaging middleware makes use of this information and can improve routing efficiency. Further extensions include zones, message queuing, streaming of multimedia data and remote method calls (RMC) built on top of the Publish/Subscribe system.

The scopes of channels are limited and organized into zones. Scoping ensures that subscribers do not receive event notifications even though they may be subscribed to relevant channels, unless the event has been propagated to a zone of which they are a member. Zones may overlap allowing publishers and subscribers to be a member of two or more zones. Typically one zone hierarchy is not enough. An organizational hierarchy is well suited for security aspects, but a hierarchy based on geographical location would be preferred for distributing context information. The concept of zones first appeared in the ECO system [8].

In case of a directed connection like between a RMC client and server stub, the used channel has exactly one publisher and one subscriber. If this constraint is known to the messaging middleware, routing can be significantly optimized. For directed connections, the advantage of the Publish/Subscribe system comes into play, when a service is moved to another location: (1) First, all subscriptions are closed at the source location. Like with Durable Subscribers in JMS, further messages will be queued. (2) Then, all objects managed by the service are serialized and sent to the target node. (3) The service is now started on the target node and initialized with the serialized object graph. (4) Finally, all

subscriptions are opened again at the target location.

4.5. Related work

Conventional middleware systems like CORBA are usually very resource intensive. They are not suitable for small devices like often found in pervasive computing and usually assume a mostly stable network environment.

Most event routing systems like Siena [4], JEDI, Keryx, Rebeca, others mentioned in [7] and many JMS implementations (`http://java.sun.com/products/jms/reference/licensees/index.html`) do not perform well in mobile environments. However, there is much research effort in this area. For example, Elvin has support for disconnected operation [10].

Modular middleware approaches that specifically target pervasive computing applications include UIC [9] and BASE [2]. JXTA (`http://java.sun.com/othertech/jxta/index.jsp`) technology is a set of open protocols that enable any connected device on the network, ranging from cell phones and wireless PDAs to PCs and servers, to communicate and collaborate in a P2P manner.

5. Summary

This paper presented my current research work in the area of Smart Digital Identities and suitable communication infrastructures for Ubiquitous Computing.

References

[1] E. Aitenbichler and M. Mühlhäuser. The Talking Assistant Headset: A Novel Terminal for Ubiquitous Computing. Technical Report TK-02/02, Department of Computer Science, Darmstadt University of Technology, 2002.

[2] C. Becker, G. Schiele, H. Gubbels, and K. Rothermel. BASE A Micro-Broker-Based Middleware for Pervasive Computing. In *Proceedings of IEEE PerCom'03*, pages 443–451, 2003.

[3] M. Boger, F. Wienberg, and W. Lamersdorf. Dejay: Unifying Concurrency and Distribution to Achieve a Distributed Java. In *Proceedings of TOOLS Europe '99, Nancy, France*. Prentice Hall, June 1999.

[4] A. Carzaniga. *Architectures for an Event Notification Service Scalable to Wide-area Networks*. PhD thesis, Politecnico di Milano, Milano, Italy, December 1998.

[5] M. D. Corner and B. D. Noble. Zero-Interaction Authentication. In *Proceedings of MOBICOM*, Atlanta, GA, September 23–28, 2002.

[6] A. Hartl, E. Aitenbichler, G. Austaller, A. Heinemann, T. Limberger, E. Braun, and M. Mühlhäuser. Engineering Multimedia-Aware Personalized Ubiquitous Services. In *Proc. of IEEE MSE*, pages 344–351, Newport Beach, CA, December 2002.

[7] R. Meier. State of the Art Review of Distributed Event Models. University of Dublin, Trinity College, March 2000.

[8] K. O'Connell, T. Dinneen, S. Collins, B. Tangney, N. Harris, and V. Cahill. Techniques for Handling Scale and Distribution in Virtual Worlds. In *Proceedings of the* 7^th *ACM SIGOPS European Workshop*, pages 17–24, 1996.

[9] M. Romn, F. Kon, and R. Campbell. Reflective Middleware: From Your Desk to Your Hand. IEEE Distributed Systems Online Journal, Special Issue on Reflective Middleware, 2001.

[10] P. Sutton and R. Arkins. Supporting Disconnectedness - Transparent Information Delivery for Mobile and Invisible Computing, 2001.

[11] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar, and J. Light. The Personal Server: Changing the Way We Think about Ubiquitous Computing. In *Proceedings of UbiComp 2002*, Gothenburg, Sweden, September 2002.