Interfacing the Real World with Ubiquitous Gateways

Christian Frank, Christof Roduner
ETH Zurich, Department of Computer Science
Clausiusstrasse 59
8092 Zurich, Switzerland
{chfrank, roduner}@inf.ethz.ch

Chie Noda, Marco Sgroi, Wolfgang Kellerer
DoCoMo Communications Laboratories Europe
Landsberger Str. 312
80687 Munich, Germany
{noda, sgroi, kellerer}@docomolab-euro.com

ABSTRACT

We describe the opportunities and challenges of using mobile phones as ubiquitous service gateways that mediate between emerging sensor technologies and networks of smart items on the one hand and existing global networks and service infrastructure on the other hand. We present an application for such a system to support users when they search for daily-life objects that they have lost or misplaced. Along with a description of the scenarios, we outline a system architecture including tagged objects, object detectors and mobile phones, and show future research directions.

1. INTRODUCTION

In our everyday life we often search for personal items such as keys, wallets, umbrellas or sports bags that we have either lost or misplaced. Emerging technologies, such as RFID systems and wireless sensor networks, properly integrated by traditional mobile infrastructure, have a great potential to allow finding these objects quickly and effectively when needed. We envision that daily-life objects will be equipped with small electronic tags that will allow detecting them via radio communication. This will allow mobile phones to *sense* objects (or act as gateways to devices with object sensing capability) and thus both record and distribute useful context information related to users and their belongings.

In this paper we present a system implementing what we call the "Remember & Find" application. The system architecture includes *tagged objects* (TOs) and *object sensors* (OSs), which are devices that can sense and identify TOs and can be integrated within mobile phones. Moreover, OSs are interconnected with the backend infrastructure through mobile phones, thus allowing remote access to objects and detection of their location. We term a mobile phone, which acts as a mediator between the OSs and the wide area network infrastructure, *ubiquitous gateway* (UG). The wide-area infrastructure can support a wide range of end-user services. For example, a UG equipped with an OS can keep track of the user's personal objects and monitor when, where and under which circumstances an object has left the UG's communication range [1]. This data can later help the user locate the object. In some cases, a user may use his UG to delegate the monitoring of personal objects. For example, smart baggage shelves can be tasked to remind a train traveller not to leave an object behind or to notify the user if the object has been taken by somebody else [2].

We consider the following use case scenarios of the "Remember & Find" application:

- **Remember Loss Context:** A UG can be used to store the context in which an object left its communication range. This includes (among others) the following data: a trace of the user's location before and after the loss event, other people present, or other personal objects carried along when the object was lost. Such information can provide the user with valuable hints regarding the location of a lost object. On the one hand, the data may help humans recall the circumstances of the loss. On the other hand, the data can be used to guide an automated object search (see next use case).
- **Find Object:** The user can send a query to many remote UGs or OSs to locate a misplaced or lost object. Queries are installed at remote gateways, e.g., colleagues' mobile phones. The user will be notified when the object is found in range.
- **Delegation of Control:** The user can delegate the tracking of an object from the personal UG to other UGs or OSs [2].
- **Lab Gate:** An OS can be installed in proximity of a student lab to record which objects leave the lab with whom. This allows intuitive and non-intrusive check in / check out processes for the equipment used in the lab.

The "Remember & Find" application is an instance of a recently emerging class of applications using heterogeneous wireless sensor networks interconnected by a wide-area infrastructure. Such applications require integrating two areas of research, namely infrastructure-supported distributed systems and energy-efficient infrastructure-less sensor networks.

Interestingly, bringing together these two areas to form a *Sensor Internet* exhibits particular new challenges. For example, in the above "Find Object" use case, scalability demands distributing a query only to a subset of all UGs or OSs present in the system. This a-priori *query scoping*, i.e., the selection of a relevant subset of sensors without ever contacting the rest, will be based on history data present in the system – data on objects, users, and their present or previous locations.

2. SYSTEM ARCHITECTURE

The above use cases require retrieving a variety of context information from sensors. The abstraction of sensor information on the UG is called a *basic event source*. When a predefined condition is detected, a *notification* is generated that contains the historical values recorded by a variety of event sources. The query service of the UG is used to specify which conditions

should be detected and what notifications should be generated. A notification can be sent to a remote user's UG or to an appropriate (global or personal) database in the backend infrastructure.

Figure 1 shows an overview of the system architecture. The backend infrastructure hosts the *global query manager*, which provides event routing and large-scale query distribution. The query distribution is based on *scope providers* which implement different heuristics for query scoping based on history data. For example, a scope provider could be based on personal or global databases containing information on past object locations and implement a heuristic that starts searching where the object was last seen. A different scope provider could employ the *location profile* (based on [3]), which is able to extract the user's most frequented locations and thus direct queries to sensors at these locations. A third example of a scope provider could be based on the *association registry*, which relates users to their objects and to other UGs in the system. Such a scope provider could direct queries for a user's object to associated UGs and to UGs of associated users, assuming that these are likely to find the object.

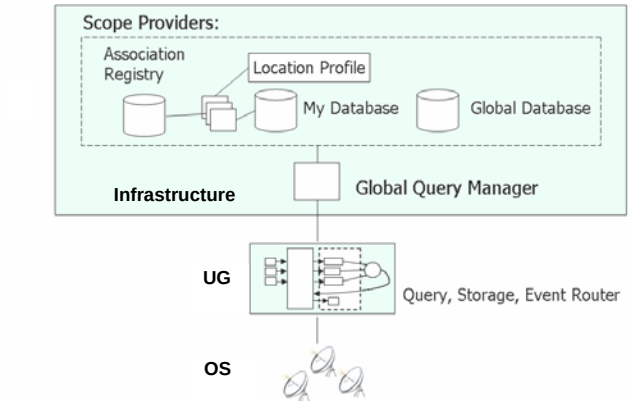

Figure 1. System Architecture

Note that scope providers need not necessarily be executed in the backend infrastructure. Rather, the middleware allows developers to configure where system components are executed. Privacy-sensitive components for example, such as the location profile or a personal association registry, can also be executed on the user UG.

3. QUERY SERVICES

We have implemented query processing services that allow for the setup of both the backend infrastructure and of the UGs involved in the above-described use cases. There are two types of queries: the UG query that specifies how an individual UG shall use its integrated OSs when processing the query (e.g., if an object should be sensed using RFID, Bluetooth, or both) and a query directed to the global query manager that specifies to which UGs a query shall be distributed, and where the resulting events shall be delivered to.

3.1 UG Query Service

As mentioned, basic event sources are components that provide a unified interface to a given kind of context data available on the UGs. Application developers can parameterize these components to generate events regarding a certain type of context information. For example the *location* source would generate an event whenever the user's location has changed (based on GPS or the current cell id). Other basic event sources used for object monitoring perform repeated polling for all tagged objects with a given frequency, while generated events include a list of discovered objects. In contrast to *public* objects that can be detected by any OS, *secure* objects can be polled only using an access key for access control and privacy protection. Here the corresponding event source component performs polling for one particular object and encapsulates the object owner's access key. An event is generated when the object is out of range for a continuous timeout interval.

The application developer may employ these basic event sources to compose a UG query: In particular, for each query *one* event source is used as a trigger for generating a notification (any event-detection functionality can be pragmatically encapsulated into a custom-implemented basic event source). The *contents* of the generated notification can include the values of any number of basic event sources, not only at the time the notification is triggered (e.g., by an event source that fires an event whenever an object is out of range) but also previous and / or future observations (e.g., of the user location). Thus the query service must store a history of values of a basic event source for a specified sliding history window. The UG query processor enables efficient memory management when multiple queries that require the same event source with different parameters are installed on a UG. Once a trigger condition is satisfied, the notification includes clips of certain data retrieved from the history windows maintained at the UG.

3.2 Global Query Service

In order to facilitate the deployment of applications in a large-scale infrastructure, our framework supports a global query manager. It enables transparent distribution of queries to multiple UGs. The global query manager relays queries to the most appropriate UGs, keeps track of active queries, and removes them upon expiration or when the searched object is found. Further, it isolates the party initiating a query from the party receiving it and vice versa. This allows effective enforcement of privacy and accounting policies. Moreover, it provides methods to monitor and control the costs incurred by querying. At the *global* query interface, the application developer may thus specify which type of *UG query* to distribute, which *scope providers* to use and a number of additional parameters, such as the maximum costs of the search.

4. FUTURE WORK

As part of current work, we are developing a search algorithm that may incorporate and integrate *any* kind of history data for query scoping, and building a prototype implementation of the middleware services we have outlined above.

5. REFERENCES

[1] G. Borriello, W. Brunette, M. Hall, C. Hartung, and D. Tangney: Reminding about tagged objects using passive RFIDs. In *Ubicomp 2004*, Nottingham, England, Sep. 2004

[2] H. Simizu et al.: Association management between everyday objects and personal devices for passengers in urban areas. Demo abstract in *Pervasive 2005*, Munich, Germany, May 2005

[3] K. Laasonen, M. Raento, H. Toivonen: Adaptive on-device location recognition. In *Pervasive 2004*, Vienna, Austria, Apr. 2004