Context Provisioning in Cellular Networks

Iris HOCHSTATTER (1), Axel Küpper (2), Michael Schiffers (2) and Lars Köthner (3)
(1) Siemens AG, Corporate Technology, Otto-Hahn-Ring 6, D-81730 Munich, Germany, E-mail: iris.hochstatter@mchp.siemens.de
(2) Department of Informatics, Ludwig-Maximilians-University, Oettingenstr. 67, D-80538 Munich, Germany, Email: {kuepper|schiffers}@informatik.uni-muenchen.de
(3) Apollis interactive AG, Steinstr. 72, 81667 München, Germany, Email: Lars.Koethner@ai.ag

Abstract

When people interact with each other, they implicitly make use of context information while intuitively deducing and interpreting their actual situation. Compared to humans, IT infrastructure cannot easily take advantage of context information in interactions. Typically, context information has to be provided explicitly. For a service to be context-aware it must be able to transparently adapt its behaviour to the user's context. However, this assumes the acquisition, refinement, and dissemination of context. In cellular networks, the context provisioning is a complex and troublesome matter. It has to be organized in an inter-organizational manner, has to be adjusted to a variety of heterogeneous service platforms, and raises issues of massive scaling and data distribution.
This paper proposes a conceptual framework for deploying context provisioning in cellular networks. By introducing different levels of abstraction, the framework eases the design, implementation, and operation of context provisioning. Furthermore, context provisioning is separated from service provisioning, i.e., both can be realized independently from each other, which supports the rapid creation and deployment of new services. In order to evaluate the proposed framework, an instant messaging service has been realized, which is also presented in this paper.

1 Introduction

In recent years, network operators and service providers have spent high efforts in search of killer applications which leverage the upcoming generation of mobile networks in order to amortize the huge investments in frequency licenses and the installation of these networks. Although these killer applications have not been identified yet, so-called killer attributes are emerging, which will play a major role in service provisioning and could therefore be an additional source of income. Today, the only killer attribute used is location, which is the key for location-aware services, such as route planning, mobile marketing, and tourist guides.
Whereas the fundamental principle of location-aware services is their adaptability with regard to the user's location, it is desirable to generalize this approach and to consider other attributes apart from location for service adaptation as well. This leads to a new philosophy in the area of service provisioning which is commonly referred to as context-awareness. Context is any information (denoted as context information in the following) that can be used to characterize the situation of a person, a place, or any other physical or abstract object, which is of relevance for a particular service (Dey, 2001). Examples of context information are the user's current activity (working, sleeping, eating, etc.), her spatial environment (indoor, outdoor, at home, at work, etc.), or the technical capabilities of the mobile device she uses. A service is regarded as context-aware if it uses context to adapt its behavior to the user's task. For example, an instant messaging service is context-aware if the message exchanges between its users (the buddies) are adapted to their current location, activities, and mood, and if the appearance of the service, e.g., the representation format of messages (text, pictures, audio, etc.), is adapted to the capabilities of the respective user's mobile device and the transport protocols of the underlying network.
A crucial point of context-awareness is Context Provisioning (CP), which comprises the acquisition, refinement, and dissemination of context information. Many research projects have been dealing with this issue in recently (Chen & Kotz, 2000). Unfortunately, most of these contributions merely consider local environments like IEEE 802.11, Bluetooth, or even single (mobile) devices. Consequently, context provisioning and respective service adaptation have been designed for a comprehensive number of users interacting in a local domain only. However, adopting the paradigm of context-awareness for large-scale environments, such as cellular networks like GSM and UMTS, puts entirely new requirements on the realization and management of context provisioning and service adaptation. Due to the global coverage area of these networks and because context-aware services are expected to be requested by a substantial amount of subscribers, scalability of context provisioning becomes an important design goal. Furthermore, cellular networks are characterized by the interaction of independent actors such as users, network operators, value-added service providers, and content providers. As a consequence, context provisioning becomes an inter-organizational matter, which, among other things, leads to questions of how to exchange and refund context information between these actors and how to protect the user's privacy.
This paper addresses some challenges of context provisioning in cellular networks and demonstrates its realization by means of a context-aware instant-messaging service. Instant messaging differs from other context-aware services in that context information must be acquired from many different sources and shared between a high number of users. It therefore puts high requirements on context provisioning and is thus an appropriate application scenario to investigate and evaluate the proposed concepts.
The paper is organized as follows: In chapter 2 we will introduce the context-aware instant messaging scenario. Chapter 3 discusses the process of context provisioning in detail by addressing a role model, the context provisioning value chain, and a plane oriented framework for modeling context-aware services. The application of these concepts to context-aware instant messaging is given in chapter 4. Chapter 5 concludes the paper.

2 Context-Aware Instant Messaging in Cellular Networks

According to Durlacher (2001) mobile instant messaging (IM) is a very popular service for i-mode users in Japan. It is expected to become equally important in other countries once UMTS services will take off (UMTS Forum, 2001) enabling mobile users to communicate anytime, anywhere using multimedia services. In addition, future IM services will be more and more context-aware as they try to adapt their functionality to the user's situation.

IM in some respects is one of the first context-aware applications broadly used. The status of a person (her presence information), displayed in a buddy list, supports communication partners to decide when and how to contact each other. And this status changes automatically even when a person does not use the application for a while. But, so far, this is the only context information used in IM systems nowadays although other types ofcontext information can be used to improve mobile IM considerably. Context-aware instant messaging (CAIM) facilitates the transparent adaptation of IM applications to the situation of the user. CAIM not only supports mobile devices but also computers traditionally used for IM as well as new sorts of IM clients, e.g., in cars or the like. According to the categories of context-aware applications by Dey (Dey, 2001) CAIM supports the 'presentation of information and services to a user' and the 'automatic execution of a service for a user'.

There are several aspects in CAIM in relation to the adaptation of the presentation of information and services to the user's needs. Obviously, and not only for IM services, the adaptation to end devices is crucial for any mobile service. With the introduction of multimedia features to mobile devices, applications need to know about the capabilities of specific devices and about network characteristics. For example, sending and receiving video streams over UMTS networks would be possible but in a GPRS network, where the user will experience rather bad quality, receiving still images or sending the video later might be preferable. However, the presentation of information should not only depend on the device used, the user's current activity has to be considered as well: when driving on a highway the user wants his IM client to read the message *"Hi Darling, I'm about to order the new Porsche. Black or red?"* aloud but when in a meeting with his boss negotiating the salary this is not desirable at all. For all mobile users the context information `location` is interesting as it changes over time and therefore the user's position is taken into account when offering services and information to a user (as described in the scenario "InviteFriends" below). The user's location can also be used as a trigger to automatically execute services. A "Blind Date" service, for example, lets a user know when there are other users around that are interested in chatting with her (e.g. based on profile information). They start a noncommittal conversation on IM and later on they can decide whether to meet in person or not.

So far we exemplified the use of context information like `location`, `activity`, `network characteristics`, `device capabilities` and `profile/preferences` in CAIM. However, enhancing IM by the use of context information yields to a new class of applications beyond "traditional instant messaging" that can only be realized using context information. In the following we will describe two example applications, the *FriendFinder* and the *InviteFriends* that offer new features in addition to the message exchange in IM.

IM is used to stay in touch with old friends and/or to find new ones. Every user builds up her own community when adding other users to her buddy list. With the use of context information there are many possibilities to extend the simple exchange of messages between users with new features.

A mobile FriendFinder service enables a user to locate her buddies. Besides the availability of a person, a user can find out whether to look out for a friend at the same bar or to send her a message because she is currently traveling in another city. The results of a request can be presented in a variety of different ways, e.g. textual, graphical using a map, etc. The location information can also be utilized to convert a position into an address or to calculate the distance between two users. This information can then be applied to sort the buddy list putting the closest buddy at the top of the list. Privacy is a critical aspect to be considered in CAIM as the users have to assent to be tracked and need to have the possibility to opt out at any time.

The FriendFinder service may be further extended to a service we call InviteFriends, which uses context information to invite friends with similar interests to special events created by a user. A user (this can also be a cinema, restaurant, bar, etc.) sets up a new event stating `type` (e.g. *cinema*), `location` and `address` (e.g. Cinemaxx Berlin, Potsdamer Platz), `date` and `time` as well as a user-defined `title` for identification. For convenience the user also adds a movie trailer and a short plot outline, then she decides what group of people she wants to invite for the event. She can either choose a group of users from her buddy list (e.g. *friends*, *family*, *work*, etc) or invite people yet unknown to her that previously affirmed their interest in blind date events. When sending out the invitation, the CAIM system determines according to profile and location information which users should receive it. Only users that are free at the specified time, interested in the type of the event, and can make it to the location will receive a message. Including reservation services and city guide applications that direct users to the specified meeting point can further enhance this scenario.

Even for the simple FriendFinder service using location information only, context provisioning is extensive as the service has to constantly track all people on the user's buddy list, transform the location information it receives to a common format, calculate the distances, and then display the list in the right order. Keeping the buddy list up-to-date at all times is a huge effort. Users are registered with different service providers and exchanging the information across provider boundaries further complicates the service implementation especially in terms of scalability and security. Complexity also increases significantly when including a variety of information from different sources.

These examples demonstrate the complexity inherently associated with context awareness, especially with context provisioning. The range of context information is very broad and may cover spatial, physical, network, device, or any other aspects. Consequently, data for deriving context information is delivered by a broad range of sources, for example sensors, devices, or databases, and therefore differ tremendously in data formats and quality. Furthermore, the CAIM application scenario assumes the cooperation between different users, operators of cellular networks, and service providers to obtain all required context information, thereby emphasizing the inter-organizational character of context provisioning.

3 Context Provisioning

Context-aware services (CAS), e.g., the IM services introduced in the previous section, assume the availability of context information. A service provider wishing to offer CASs does not only have to care for the realization of the services, but he must also deal with the question how to create or how to attain the associated context information. All steps needed to be carried out in order to create or attain a context information are subsumed under the term *context provisioning (CP)*. CP is a rather complex process consisting of many sequential and parallel sub-processes. In order to identify, structure, and modularize CP we introduce the concept of a value chain that covers all sub-processes happening between sensors and a CAS. Figure 1 shows the general structure of such a value chain.

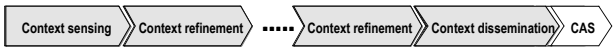

Figure 1: The Value Chain of Context Provisioning

The context of an entity, e.g., the user of an IM service, may be derived from very different *context sources*, e.g., positioning services like GPS for obtaining the user's current location, or databases containing her buddy list. Capturing data from these context sources is a process we call *sensing*. However, sensing merely provides the "raw material" of context, which is referred to as *low-level context information* (Dey, 2001), and which often is not interpretable by the requesting CAS. Usually, one or several steps of *refinement* have to be performed in order to derive *high-level context information* as required by the respective CAS. Among other things, refinement comprises the transformation between different formats of representation (e.g., from GPS coordinates to street names and numbers), the extension of context information with quality attributes (*Quality of Context*, QoC), e.g., in order to express its accuracy (Buchholz et al. 2003), or the combination of context information to derive another one (e.g., calculating the distance between two users of an IM service session). After the refinement process, context information must be delivered to CASs, a process we call *dissemination*. Generally, dissemination can be subdivided into a *pull* and a *push mode*. Using the pull mode, context information is requested during CAS usage. It can then be further classified into a *polling* and a *caching mode*. In the former case, context refinement is just activated on demand, whereas in caching mode, it is executed in a proactive manner. The push mode is useful to realize context-aware push services, e.g., an SMS-based IM service that alerts a user when one of her buddies comes into her close vicinity. Thus, context dissemination represents the interface between a CAS and the associated CP and comprises different ways in which context information is accessed.

3.1 Roles in Context Provisioning

As demonstrated in the previous section, context information may be derived from very different and heterogeneous context sources like sensors in the mobile device and in the environment, tags and beacons, or databases, to name only a few. However, in cellular networks it is unlikely that providers of CASs will own and autonomously operate all context sources being of relevance for adapting their CASs. Rather, it is to be expected that these sources will be operated by different (and competing) actors, and a service provider wishing to offer a CAS has to negotiate conditions about the delivery of context information from these actors. Among other things, these negotiations need to cover QoC considerations, terms and conditions of payment as well as of usage of private data. Thus, a non-trivial CAS can only be realized in an inter-organizational manner, where different actors interact with each other in order to acquire, distribute, and process context information. Unfortunately, inter-organizational aspects of context provisioning have not been covered so far by research activities in this area.
In order to further investigate these inter-organizational aspects we propose a role model for CASs (see figure 2). In this model, an actor denotes an individual, an organization, department, or any enterprise either offering services to other actors, or consuming services from other actors, or doing both. From a technical point of view, each actor autonomously operates and controls its own technical domain, consisting, for example, of a network infrastructure, a server farm, or only a single mobile device. An actor may adopt one or several *roles*. A role represents a certain field of activity of an actor and comprises a well-defined set of tasks an actor adopting this role has to fulfil. Relationships between these roles reflect technical co-ordinations, such as protocols, interfaces, agreements concerning the pricing and quality of a particular service, and mechanisms for observing the fulfilment of these agreements.

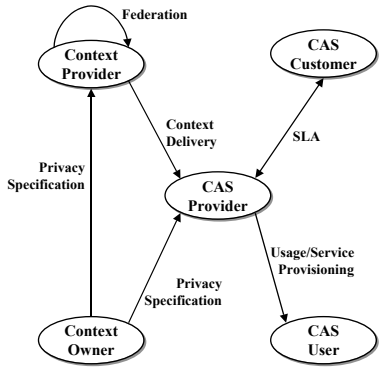

Figure 2: Context Provisioning Role Model

Figure 2 shows our approach of a role model for CASs, which is independent of a particular network technology and which highlights the special problems and tasks of context-awareness. The central role of our model is the *CAS provider*, which creates and deploys CASs and offers and sells them to a *CAS customer*. The CAS customer interacts with the CAS provider in order to negotiate the terms of CAS usage on behalf of one or several *CAS users*. The CAS provider obtains context information for service adaptation from a *context provider*, which is usually the operator of context sources. For example, a context provider may be the operator of a cellular network, which tracks a CAS user and delivers that user's current location to the CAS provider. For many CASs, especially for IM services, it would be desirable that a CAS user has access to context information, which is related to another actor. From a security point of view, this is a very sensitive matter, because an actor must always have control about the accessing and processing of her context by other actors. It is therefore inevitable to establish the role of a *context owner*, which represents an entity context is related to and which is able to specify access restrictions regarding her context.

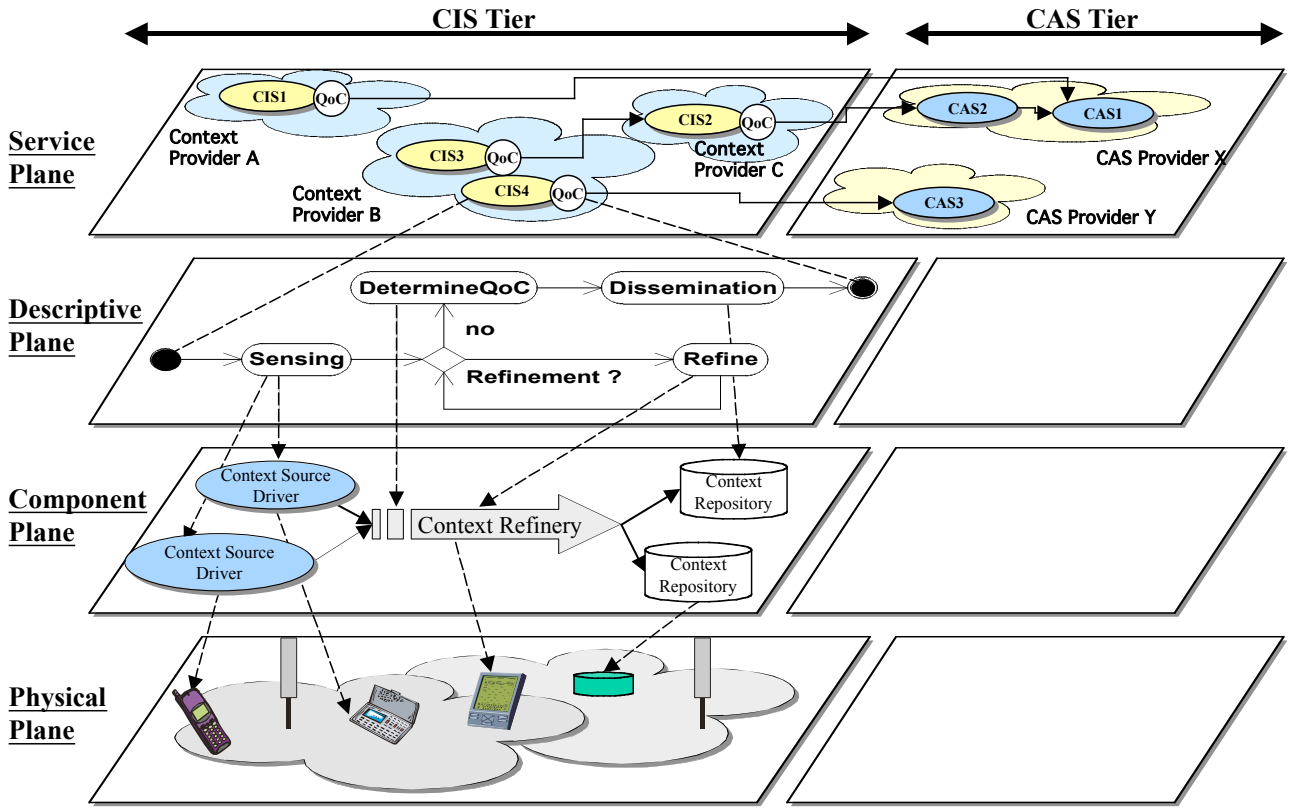

Figure 3 A Framework for Context Provisioning

3.2 Framework for Context Provisioning

In cellular networks, context provisioning is a rather complex and troublesome matter. As indicated by our role model, various actors may participate in the provisioning of context information and CASs and need to be coordinated. Furthermore, there is a variety of partly proprietary and partly standardized service platform, e.g. the Short and Multimedia Message Service (SMS, MMS), the Wireless Application Protocol (WAP), i-mode, and Java-based solutions, CP needs to be adjusted for. Cellular networks cover huge geographic areas, usually on a national basis, and serve a huge number of subscribers. Thus, introducing context provisioning would raise issues of massive scaling and data distribution.

To cope with the aforementioned challenges of cellular networks, we propose a plane-oriented framework that eases the design, implementation, and operation of context provisioning. Every plane in the framework reflects the value chain of context provisioning from a certain level of abstraction and granularity respectively. The planes are called *service*, *descriptive*, *component*, and *physical plane*, and each plane hides the aspects of its underlying planes. Each of them is subdivided into a *context information service* (*CIS) tier*, covering the provisioning of context information, and a *CAS tier*, covering the provisioning of CASs. The distinction between different planes on the one hand and between CIS and CAS tier on the other, enables the reusability of software, hardware, and specifications, a seamless integration into existing service platforms and cellular-network infrastructures, and a rapid service creation and deployment of new value chains and services. Although the aspects of a plane can be treated independently from those of other planes to a certain degree, dependencies must be defined between the planes when realizing the system. Figure 3 depicts the framework.

On the service plane, context information, such as location, distance or temperature, is modelled as the deliverable of a CIS, thereby hiding all aspects of context provisioning. The service plane addresses the relationships between CASs and CISs as well as all administrative aspects in terms of actors and roles participating in context and service provisioning. Additional care has to be taken to provide information about the QoC and terms of usage, for example the costs or a list of actors authorized for usage. Furthermore, constraints concerning the required QoC and the way context information has to be requested (e.g., push or pull) may be specified. Note that the single steps of the actual context provisioning, their specification, the specification of the involved infrastructure components, and the used technologies are not visible on this plane.

Looking at the service plane as the starting point for a "roadmap" to modelling context-awareness we focus on understanding the requirements, the concepts, and the terms of operation of CISs on a very abstract level. The center of attention is on the *what*: what are the roles, what are the concepts and terms, and what are general operations? On the descriptive plane, we will address the *how* instead: how does a CIS generate context information? How is the functionality a CIS provides on the service plane mapped onto value chains.

Consequently, the descriptive plane contains a formal specification of the CP process. Functionally describing a CIS means mapping it onto one or several value chains on the descriptive plane, depending on the requirements for a concrete context information (e.g. QoC). This mapping is not unique as there are obviously several ways to "arrange" or "serialize" the refining steps.

The descriptive plane is completely process driven in that it requires a formal specification for context sensing, context refinement, and context dissemination. For context sensing this includes a description of context information, the quality these information should have, and the way they can be accessed. The relevant information are selected and aggregated during context refinement.
On the component plane, context sensing, refinement, and dissemination need to be mapped onto one or several components. Context sensing is executed by a *context source driver (CSD)*, context refinement by a *context refinery*, and, finally, context dissemination is done by a *context repository*. These components are still more generic than physical devices. For example, CSDs may represent a broadly heterogeneous spectrum of context sources. Although a CSD has to control a context source according to its special characteristics, it should hide them from other components by uniform interfaces, which are independent of the respective type of context source and which are common for all types of context sources. The component plane typically assumes the existence of a common middleware architecture, on which components are executed and which provides a facility for remote invocations. In cellular networks, the Java platform together with CORBA/RMI is the first choice for that. As the CIS repository represents an interface between CIS and CAS tier, it might offer additional interfaces, for example in order to support context-aware speech, SMS, MMS, WAP, or i-mode services.
The physical plane links to the real world and reflects the essential physical nodes, communication links between nodes, and the resulting topology of the cellular network. Generally, a node may represent any kind of hardware with a computation environment. For example, dedicated service nodes, as envisaged for UMTS, accommodate all the logic needed for service provisioning. In the same manner, context nodes host all components of CP. This also includes sensors, databases, mobile devices, or any other physical object being involved in context provisioning. Communication links interconnect nodes and handle the information flows between them. They may be wired or wireless.

4 Implementation

To evaluate the developed concepts we designed and implemented a context-aware instant messaging service (CAIMS). According to the framework for context provisioning we modeled the FriendFinder service top-down describing the elements of all four planes and mapping each of them onto elements of the underlying plane.

Service Plane

On the service plane, we abstractly model the FriendFinder service described in subsection 2. On this level of abstraction we do neither consider how context information may be derived from others nor do we address where the context information will come from. Hence we are independent of particular context providers and components and the service description can be used for any FriendFinder-like service at any CAS provider. Depicted on the left hand side of figure 4 is the CIS tier of the FriendFinder's service plane with the three context information items used in the service, BuddyList, distance, and profile. Context information is always related to a context owner (see subsection 3.1), in our example typically a user and her buddies. It has to fulfill certain quality constraints, e.g. precision and up-to-dateness. On the right hand side, the CAS tier shows the FriendFinder service itself. We modeled the FriendFinder service with UML use cases though other modeling techniques can be applied. We will not show the use cases in detail but use the description of the service given in section 2. The InviteFriends application relies on the FriendFinder service as stated in section 2.

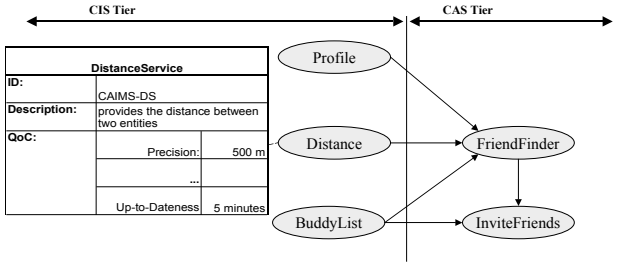

Figure 4 Service Plane FriendFinder

Descriptive Plane

Figure 5 illustrates the distinction in CIS and CAS tier for the descriptive plane. The context information `distance` from the service plane cannot be directly sensed and is therefore split into the two parts, `location` of user A and `location` of user B. The two context information have to be sensed at first, the distance is then calculated in a refinement step, and is finally made available for the services to use. The CAS FriendFinder on this level of abstraction is represented in UML sequence diagrams (cp. Figure 5, right part) to model the process of service provisioning.

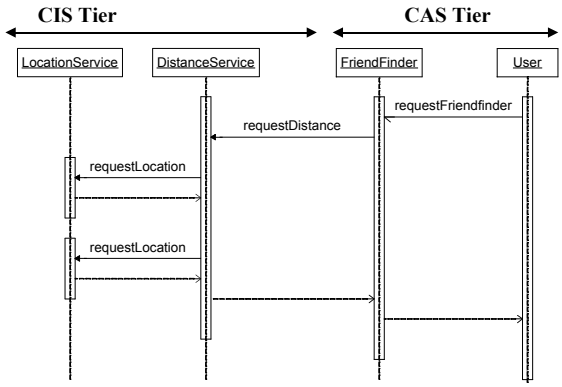

Figure 5 Descriptive Plane FriendFinder

Component Plane

The component plane's CIS tier shows the actual system components that sense and refine context for the FriendFinder service (see figure 6).

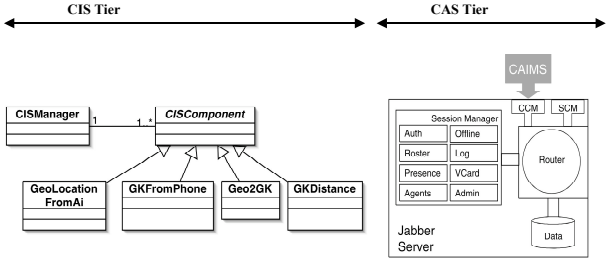

Figure 6 Component Plane FriendFinder

Context source drivers (CSD) in our example are components that connect to a variety of sensors. Context

refineries are described specifying one or more input and the output context information. There is a central authority *CISManager* where all components are registered, this manager also acts as CIS repository. For the FriendFinder service, location information for the two users is needed. It is either requested from mobile network operator Vodafone D2 or gathered from the mobile device itself (using the Location-Aware Mobility Architecture, see Küpper *et al*, 2003). In our prototype implementation, there are two context refineries that calculate the distance between two users, the *GKDistance* component uses location information in Gauß-Krüger coordinates, the *GeoDistance* component calculates `distance` from location information in WGS 84 geo coordinates. When the FriendFinder CAS requests the distance between two users, the CISManager has to find a tree of sensing and refinement components that can fulfill the request (see figure 7). The search algorithm is doing a breadth-first-search over his database of registered components.

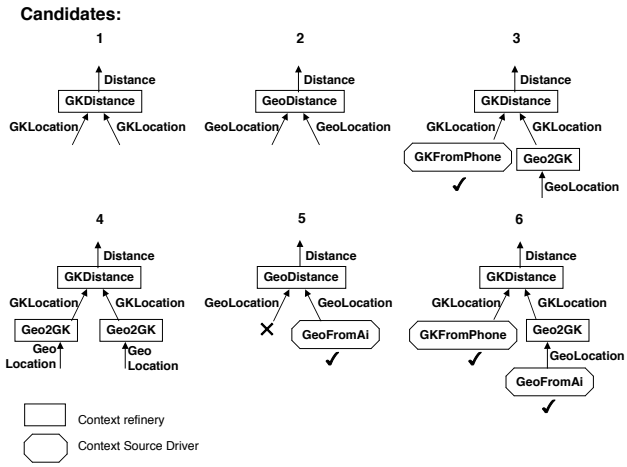

Figure 7 Finding a Tree of Refinement Components in CAIMS

In the first run, every component that outputs the requested context information is added to a list of candidates. In the FriendFinder scenario, these are the two components `GeoDistance` and `GKDistance`. When finding a CSD, which does not need input information (marked with a V in figure 7, e.g. step 6), the algorithm terminates. Otherwise in the next run, components are searched that output the context information needed as input for the first component. Like this, all child elements of the same height are added in one run. If there is no component present in the system that can deliver the information, e.g. `GeoLocation` for user A (marked with an X in figure 7, step 5), the candidate tree is discarded. In our example, the location information for one user can be sensed in Gauß-Krüger format at the mobile device, for the other user it has to be requested in geo format from the mobile network operator. Thus, another refinement component is needed first to transform the WGS 84 format to Gauß-Krüger coordinates, then the distance can be calculated and returned to the FriendFinder CAS.
CAIMS was integrated into the open source IM platform Jabber (Adams, 2002). Jabber is an XML protocol for the real-time exchange of messages and presence; it is an open and extensible system. Everyone can run his own server and extend its functionality. The CAS part in the component plane consists of standard Jabber components plus the additionally implemented CAIM service (see figure 6). We implemented CAIMS using Jabberbeans, a Java library to enable applications to interact with Jabber servers. The CAIMS component connects to the Jabber server as a Jabber client. In this case it communicates with the Jabber Session Manager directly and can therefore use its functionality e.g. presence and message relay.

We do not go into details of the physical plane as our CAIMS utilizes existing physical components, e.g., of the mobile network operators' infrastructure, but does not influence which physical nodes or communication links are used. When mapping components from the component plane to existing objects, performance considerations tell the provider where to place each component and which components have to be replicated.

5 Conclusion and Outlook

Context provisioning in cellular networks is a complex process where multiple providers are involved. To model context-aware services and context provisioning in this environment, we proposed a role model and an adequate framework. These abstractions help us to separate concerns and therefore support the reuse of system parts. In this paper we applied the framework to context-aware instant messaging. However, due to the separation of context and service provisioning, the framework is applicable to context provisioning for cellular networks in general.
To support context provisioning in multi-provider environments, there are many more issues that have to be dealt with. Important topics for the future are the exchange, interpretation, and composition of context information, as well as the quality of context information and how to protect the user's privacy.

6 Acknowledgement

The authors wish to thank the members of the Munich Network Management (MNM) Team for helpful discussions and valuable comments on previous versions of the paper. The MNM Team directed by Prof. Dr. Heinz-Gerd Hegering and Prof. Dr. Claudia Linnhoff-Popien is a group of researchers at the University of Munich, the Munich University of Technology, and the Leibniz Supercomputing Center of the Bavarian Academy of Sciences. For further information see http://wwwmnmteam.informatik.uni-muenchen.de.

7 References

Adams, DJ (2002), *Programming Jabber*, O'Reilly, Sebastopol.

Buchholz, T., Küpper, A. and Schiffers, M., *Quality of Context: What it is and why we need it*, in Proceedings of the 10th HP–OVUA Workshop, Geneva, Switzerland, July 6 - 9, 2003.

Chen, G., Kotz, D., *A Survey of Context-Aware Mobile Computing Research*, Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, Nov. 2000

Dey, A. (2001), *Understanding and Using Context*, Personal and Ubiquitous Computing Journal, Vol. 5 (1), 2001, 4-7.

Durlacher (2001), *UMTS Report: An Investment Perspective*, Research Report, Durlacher.

Küpper, A., Fuchs, F., Schiffers, M., Buchholz, T. (2003), *Supporting Proactive Location-Aware Services in Cellular Networks*, in Proceedings of PWC 2003, Personal Wireless Communications, Venice, Italy, September 23-25, 2003.

UMTS Forum (2001), *The UMTS Third Generation Market – Phase II: Structuring the Services Revenue Opportunities*, UMTS Forum Report #13.