Modeling Context Information for Realizing Simple Mobile Services

Gregor Broll, Heinrich Hußmann, George N. Prezerakos, Georgia Kapitsaki, Stefano Salsano

Abstract—**The emerging technology of mobile services is gaining on importance nowadays and various applications are being developed. Context-awareness constitutes an essential part of mobile services, as it can boost the provision of new personalized services (e.g. location based services). In this paper, the concept of a UML-based context model for the Simple Mobile Services (SMS) project is presented. The SMS Context Model focuses on modeling context information in a structured and comprehensible way in order to facilitate the development, provision and operation of context-aware mobile services. Its three-level approach (meta, model and instance) offers different levels of abstraction and can be used as a basis for transforming the resulting model to various implementation languages.**

Index Terms—**Context model, Simple Mobile Services (SMS)**

I. INTRODUCTION

AS a result of the rapid development of Mobile Computing, the widespread usage of mobile services has been increasing during the last years. Today, mobile services can be used for all kinds of tasks like e-commerce, mobile banking, remote maintenance or location based services in particular. From the providers' perspective, mobile services offer great opportunities to bring information, business and commerce to a wider group of customers.

Despite their usefulness and advantages for providers and consumers, mobile services are still not as advanced, widespread and established as services in Desktop Computing. Apart from the obvious constraints of their target platforms - mobile devices with little memory, low processing power, small screens or fiddly joysticks and keys - that often

G. Broll and H. Hußmann are with the Media Informatics Group, LMU, Munich, Germany (phone: +49-89-2180-4655; fax: +49-89-2180-4652; e-mail: {gregor.broll, heinrich.hussmann }@ifi.lmu.de).

G. N. Prezerakos and G. Kapitsaki are with the School of Electrical & Computer Engineering, National Technical University of Athens (e- mail: {prezerak, gkapi}@telecom.ntua.gr).

S. Salsano is with the Dipartimento di Ingegneria Elettronica (Electronic Engineering Department), Università degli Studi di Roma - Tor Vergata, Rome (e-mail: stefano.salsano@uniroma2.it).

decrease their usability, mobile services are often hard to set up, hard to find, hard to use and hard to trust. Without standardized guidelines and paradigms for their development, most mobile services have been created from scratch.

In this context, the goal of the Simple Mobile Services (SMS) project [1] is to create innovative tools enabling a new class of services, addressing the specific needs of mobile users and enabling individuals and small businesses to become service providers. The vision of this project is to support the simple authoring and standardized provision of services that are easy to find, easy to use, easy to trust and easy to set-up.

An important aspect of mobile services is context-awareness whose significance is emphasized by the fact that location based services are their most prominent representatives. Context-awareness is one of the most determining factors that distinguish services in Mobile Computing from services in Desktop Computing. Mobile devices are used in a highly dynamic, mobile and personalized context that provides an abundance of information. Mobile devices can access and use this information in order to adapt themselves to it, outbalance technical constraints and enable new functionalities or even new applications.

Following its vision, the SMS system has to be able to model, manage and provide context information in order to facilitate the authoring, provision and usage of simple, context-aware services. This paper presents the concept for the SMS Context Model as an integral part of these processes.

The paper is structured as follows: The next chapter presents related work concerning the SMS Context Model. Chapter III illustrates the usage of SMS with a reference scenario and chapter IV briefly introduces the SMS approach to service authoring. Chapter V explains the requirements and conceptual constraints for the SMS Context Model which is described in more detail in chapter VI. Finally, chapter VII concludes this paper.

II. BACKGROUND AND RELATED WORK

Probably the most common and systematic definition of context comes from Dey and Abowd [2]. They analyzed and compared several previous definitions which describe context information by using synonyms or enumerating examples. Dey and Abowd developed their own definition of context as a summary and abstraction of the previous ones and emphasized the ubiquitous character of context information

and its relevance to the described entity: „*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.*" [2].

For the SIMPLICITY project [3] – a predecessor of the SMS project – Rukzio et al. [4] present an approach that builds upon existing methodologies and recommendations from international standards organizations in order to model common entities such as people, locations, devices, services and their key characteristics. Modeling context in SIMPLICITY is split into two phases: The analysis phase uses the UML-based Common Information Model (CIM) [5] standard to model key entities, their properties and relations. The resulting *Simplicity Information Model* is a high-level representation of the system and the foundation for the SIMPLICITY context model in the implementation phase. In this phase, the abstract information model serves as a blueprint for the more detailed modeling of context information using standards like the 3GPP Generic User Profile [6] in order to provide a framework that describes the syntax of specific context models. This approach clearly separates the abstract modeling of context and its more detailed implementation for the development of applications. However, it does not exploit all features of CIM for modeling context.

ContextUML [7] is a UML-based model for the specification and model-driven development of *Context-aware Web Services (CAS)*. It provides meta-models for context, services and context-aware mechanisms that associate both with each other. The meta-model for context information is centered around the *Context* class that represents generic context information. It is further sub-classed into *AtomicContext* (simple, low-level context, directly provided by a context source) and *CompositeContext* (high-level context, aggregates multiple atomic or composite contexts). The classes *ContextSource, ContextService* and *ContextServiceCommunity* model the resources from which context information is retrieved. Although this model is integrated into a complete meta-model for implementing CAS, it is restricted to them and lacks grounding in real application scenarios.

Context information can be complex and vulnerable to different errors such as incorrectness or ambiguity. Buchholz et al. define *Quality of Context (QoC)* as "*any information that describes the quality of information that is used as context information. Thus, QoC refers to information and not to the process nor the hardware component that possibly provide the information*" [8]. For a more precise understanding of QoC, Buchholz et al. refine the term by outlining it with its most important parameters – precision, probability of correctness, trust-worthiness, resolution and up-to-dateness. QoC-parameters can be used to agree on the allocation and reception of context information of a certain quality between context providers and users, the selection of appropriate context providers, the adaptation of context refinement and the definition of context-related privacy policies.

III.   FINDING A GOOD RESTAURANT – A SMS REFERENCE SCENARIO

In order to illustrate the vision and the ideas behind SMS, the following scenarios illustrate the usage of simple mobile services from the end users' point of view.

Simon is attending a conference in Rome. On a free evening he has a very good meal at a restaurant and wants to share this experience with other people. The city of Rome has recently adopted the SMS system to set up and provide different context-aware mobile services for its citizens and tourists. Simon decides to use the "Recommend"-service from his mobile phone. He authorizes the service to determine his location using the GPS-functionality of his mobile phone. The SMS-service shows Simon his position on a map and highlights different sights, shops and restaurants in his vicinity. He clicks on the icon representing the restaurant and opens a form with different options for his recommendation including overall satisfaction, type and quality of food, price level, service, atmosphere, etc. Simon quickly fills out the form, takes a photo of the restaurant's garden and uploads both to the server from where other people can use it.

Some weeks later, Nina is on a vacation in Rome. After she has been using a SMS-service to guide her private sigh-seeing tour, she now uses another service to have a good restaurant recommended. She applies different filters recognizing her location, the available restaurants in her vicinity and her personal food preferences. The service is allowed to use this information from her personal profile on the mobile phone. The service suggests 3 restaurants and Nina selects them depending on the average of received recommendations in order to see more details. As she selects the restaurant that Simon has enjoyed so much, she reads his recommendation and looks at the picture of the restaurant's garden. Excited by the recommendation, Nina decides to give it a try and the service shows her the way to a great dinner.

IV.   THE SMS APPROACH TO SERVICE AUTHORING

Fig. 1 provides a graphical representation of the SMS approach to service authoring. A detailed analysis of this approach is reported in [9].

SMS is considering two levels of service authoring; one is directed to the expert programmer or software designer and is called *high level modeling*. The second level is called *Authoring Wizard level* and is directed to people with minimal technical expertise. At both levels, it is possible to compose a service by putting together and customizing existing *service elements* or *component services*. The two levels differ for the different representations of these service elements and for the different expressiveness of the combination mechanism.

The SMS high level modeling is based on UML. An UML profile for SMS has been defined, the *service components* are

represented by UML components and their interfaces are represented by UML interfaces. Composing services at this level is equivalent (in terms of capability) to using a programming language, allowing a great freedom in realizing services. The Authoring Wizard level does not require the knowledge of UML nor any programming capability. On the other hand the combination of service elements is more rigid as it follows predefined templates. There are also limitations on how the components can be combined.
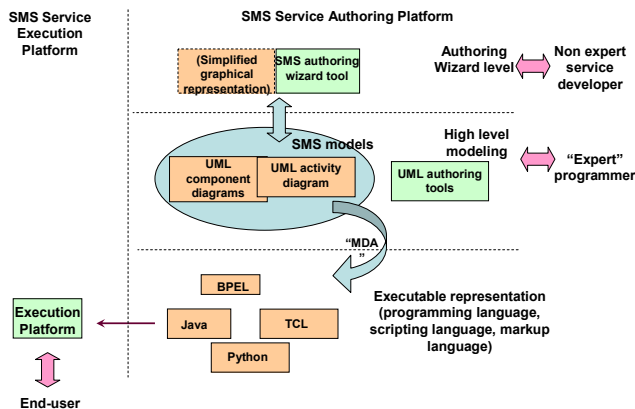

Fig. 1. The SMS approach to service authoring

This paper mainly deals with aspects of the high level modeling. At this level, the UML component services can be composed using a UML activity diagram, where each activity corresponds to a component operation (more details in [9]). Knowledge of UML and technical details about the SMS platforms is required to compose services at this level. The UML representation is meant to automatically generate an executable representation into a programming, scripting or markup language.

This approach assumes the availability of a library of basic component services that are not developed using the SMS service authoring platform. The SMS authoring platform can build end-user services out of the basic building blocks, or it can build additional complex component services which can recursively be used in further composition processes.

The SMS component services need to consider context information in their behavior. This may happen explicitly, i.e. a service designer can build a service where a decision is taken depending on some context information. An example of such a service could be: "If the user is on vacation send him a picture and a description of interesting nearby monuments". In other cases the context adaptation is done implicitly by a basic component services. For example a basic component service could provide a notification of a message to a user selecting the most appropriate means of communication (SMS, e-mail, voice call with automatic text-to-speech announcement) depending on the user context. In both cases there is the need to model context information in UML, in order to understand which information is available to adapt the service or to take explicit decision within a given service logic.

V. REQUIREMENTS FOR A SMS CONTEXT MODEL

In order to integrate context information into the authoring of services and the modeling of their context-aware behavior, a context model for SMS has to accommodate its approach to service authoring. It also has to be able to model context information on a high level using UML. Apart from this conceptual aspect, it has to be possible to break this high-level model down to a more specific modeling and typing of context information that can be used by services at runtime.

Keeping in mind the definition of context from Dey and Abowd the first step in a bottom-up approach to such a context model is to determine the main entities that should be described. Among them are usually users, devices, services and networks. By examining them and their characteristics, different categories of context information can be identified that are relevant for providing context-awareness: user (identity, preferences …), device (hardware, software …), network (capacity, data rate…), security/privacy (authorization, trusted entities…), social context (business, leisure …), physical context (temperature, traffic…), or payment (prepaid, account information…).

The SMS Context Model not only has to accommodate to the different levels of the SMS authoring process, but also to different constraints that originate from the versatile and diverse nature of context information itself. It is usually gathered from single or multiple sensors and thus often ambiguous or incomplete. Often enough, sensor data can't be used directly in context-aware applications but have to be translated, transformed or refined first.

The SMS Context Model must consider its usage in context-aware services and the SMS framework that helps building them. For a systematic approach it is necessary to deal with the richness, structure and quality of context information as well as its processing, diversification and evolution. For that purpose the SMS Context Model classifies context information using the two dimensions of quantity and quality, which can be mapped to and described by the two pairs of opposite terms atomic/composite and low-level/high-level context information. These terms are often used for the basic and informal categorization of context information (e.g. [7]).

For the SMS Context Model, the differentiation between atomic and composite context information is the following: Atomic context information is a single context information that is acquired from one source (e.g. sensor, database, explicit user input …). It can exist by itself and often has an absolute, technical meaning without any influence from the outside (e.g. GPS-coordinates). For composite context information on the opposite, context information is always interpreted in relation to other context information or conventions. It comprises and relates multiple pieces of context information whose combination has a meaning of its own. This new meaning is defined through the combination of single parts rather than by the single parts themselves. Composite context information is based on multiple atomic

and/or composite context information which is gathered and processed by different mechanisms (e.g. collection, history, inference ….). In the process, the mechanisms can keep the compound nature of composite context (e.g. a user profile simply collects different context information) or create a new, single, yet complex context information, e.g. behavior which is based on a history of many similar, temporally ordered context information.

The level of complexity or abstraction of context information can be rated on a scale between low-level and high-level using *Quality of Context (QoC)* parameters. Different mechanisms can change the level of complexity of atomic and composite context information to a higher, lower or even the same level. Examples for such mechanisms are: refinement (high level: traveling to x, lower level: at the airport, lower: needing to check-in, lowest: sensed at queue), translation/transformation (e.g. from GPS-NMEA data to a street address) or inference.

This categorization along the dimensions of atomic/composite and low-level/high-level context describes context information at different levels and keeps the model rather simple. Table I shows how different context information can be classified using this differentiation:

TABLE I: Relationship between low/high level context and atomic/composite context information

	Atomic	Composite
Low level	GPS positioning data, time in milliseconds	user profile (collects different values e.g. name, address, age, gender, …)
High level	street address, time in the hh:mm:ss-format	behavior (based upon a history of single context information)

VI. The SMS Context Model

The main task of the SMS Context Model is to model context information in a structured and comprehensible way in order to facilitate the development, provision and operation of context-aware mobile services. The previous section outlined the requirements and constraints that influenced the design of the SMS Context Model, which is also inspired by the approaches of SIMPLICITY and ContextUML for several reasons: SIMPLICITY modeled context information in two distinct phases starting from the abstract definition of a generic, high-level information model of key entities and proceeding to its translation to a concrete context model. The SMS Context Model adopts this top-down-approach and combines it with the more sophisticated and structured approach of ContextUML. Its context meta-model already meets important requirements of the SMS Context Model: It includes the useful distinction between atomic and composite context and its integration into a complete meta-model for context-aware services is a great example for the integration of the SMS Context Model into the SMS framework for building simple mobile services.

Similar to SIMPLICITY and ContextUML, the SMS

Context Model (see Fig. 2) uses UML to model context information. However it is more elaborate and includes 3 different levels of abstraction (meta, model and instance). The resulting UML model can be used to derive context models using other languages, e.g. XML-related standards like 3GPP. Advantages of this approach (especially for SMS) are:

- Using UML for modeling context information and thus being able to associate and integrate it with the UML based modeling of services, workflows, etc. in SMS. This feature is similarly supported by ContextUML, which embeds its context model into a meta-model for CAS.
- The clearly structured UML model allows the modeling of all context information for an application on different levels of abstraction from meta to instance level.
- While the UML model is rather conceptual, it can be used as a blueprint or template from which to derive descriptions of context using other languages that are closer to the actual implementations of applications and services. This approach is similar to SIMPLICITY's separation between a more conceptual information model and a more detailed context model.
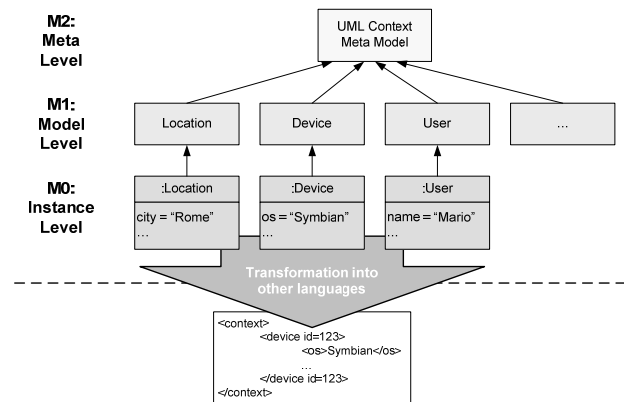

Fig. 2. The 3 levels of UML context modeling for SMS

The *Meta Level (M2)* of the SMS Context Model provides a meta-model for context information itself (see Fig. 3) which is missing in SIMPLICITY. It defines generic entities for the modeling of context, its structure, properties and relations on a high level of abstraction, independent of concrete context information (e.g. location, device …). The context model on this level is designed after the context meta-model of ContextUML which already meets many of the requirements of the SMS Context Model. Thus the SMS approach adopts the classes for Context, AtomicContext and CompositeContext from ContextUML, but simplifies the modeling of context sources and adds its own class for describing quality of context parameters (see Fig. 3). This *QualityofContext* class associates context information with appropriate Quality of Context-parameters that identify it as either low-level or high-level context.

The Meta Level represents the most abstract level on which context information and applications/services can be associated, similar to the approach of ContextUML. This

model can be further extended to include other aspects of context and meet requirements for SMS and its applications. It could provide more elaborate models, e.g. for additional context properties, user interfaces, Quality of Service-parameters or context sources.
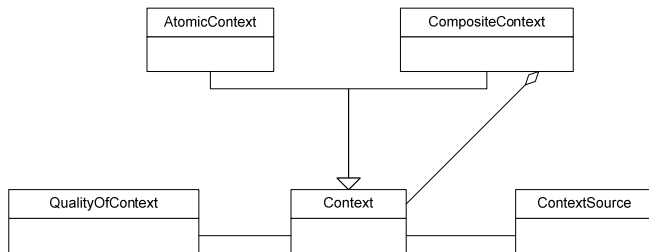

Fig. 3. The SMS Context Model meta-model for context information

The *Model Level (M1)* concretizes the Meta Level and includes different characterizations of atomic and composite context information like location, user or device. Its main purpose is to define and lay out a model of classes and subclasses of context information used for an application/service and model its typical properties on an abstract level. The classes included in Fig. 2 are the ones already included in the SIMPLICITY information model. The analysis of the SMS example services will lead to a more complete (but still extendible) model. As the context information modeled on this level uses the entities defined in the Meta Level, it can be used as a benchmark to check its validity.

Picking up the reference scenario from chapter III, Fig. 4 shows a UML model of context information that is relevant for finding a good restaurant. On the Model Level, the involved contextual entities could be User Profile, Food Preference, Restaurant, Location, Dinner Option and Diner Recommendation, along with their different attributes and properties. Diner Recommendation for example is a composite context information that consists of several Dinner Options, which are derived from the atomic contexts Location, Restaurant and users' Food Preferences. A service that uses this context model, could match the food preferences and the location of a user against the address and the type of food of several restaurants in order to provide a recommendation that is close to the user's preferences.
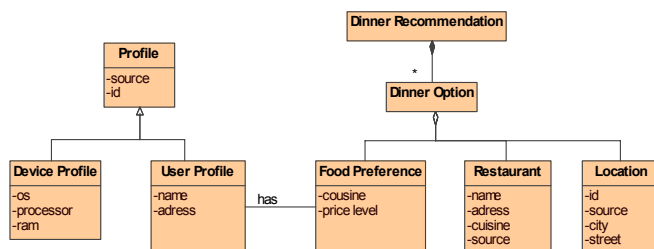

Fig. 4. Model for different context classes

Finally, the *Instance Level (M0)* comprises concrete instances of context information from the model level and

gives individual context an "identity" by specifying values for the abstract model. On this level, Nina from the reference scenario is guided from her location near the Coliseum in Rome to the restaurant "Da Luigi" where she can eat her beloved Spaghetti Carbonara. As this level contains all context information and values that are used in an application or service, it can act as the foundation for a transformation of this information into another programming language that is more suitable for an actual implementation than UML, e.g. XML.

VII. CONCLUSION

This paper outlined the concept for a context model that accommodates the SMS approach to service authoring. The SMS Context Model is designed to model context information on different levels of abstraction from high-level UML modeling to their translation into concrete context models that can be used by services at runtime. As the basic concept of the SMS Context Model is promising, future work will enhance and evolve it into an integral component of the SMS framework.

REFERENCES

[1]   Simple Mobile Services (SMS) website: www.ist-sms.org
[2]   A. K. Dey, and G. D. Abowd, "Towards a better understanding of context and context-awareness," *GVU technical report GIT-GVU-99-22*, College Computing, GA Institute of Technology (1999).
[3]   Simplicity website. www.ist-simplicity.org
[4]   E. Rukzio, G. N. Prezerakos, G. Cortese, E. Koutsoloukas, and S. Kapellaki, "Context for Simplicity A Basis for Context-aware Systems Based on the 3GPP Generic User Profile International Conference on Computational Intelligence," ICCI 2004, Istanbul, Turkey, pp. 17-19 December 2004.
[5]   Distributed Management Task Force. Common Information Model. http://www.dmtf.org/standards/cim/
[6]   3rd Generation Partnership Project. Generic User Profile – architecture. Technical specification of Technical Specification Group Services and System Aspects, Version 6.3.0. 2004
[7]   Q. Z. Sheng, and B. Benatallah, "ContextUML: A UML-Based Modeling Language for Model-Driven Development of Context-Aware Web Services," The 4th International Conference on Mobile Business (ICMB'05), IEEE Computer Society. July 11-13 2005, Sydney, Australia.
[8]   T. Buchholz, A. Küpper, and M. Schiffers, "Quality of Context Information: What it is and why wee need it," *Proceedings of the 10th International Workshop of the HP OpenView University Association (HPOVUA'01)*, Vol. 2003, Geneva, Switzerland, July 2003
[9]   S. Salsano (ed.) "Initial system architecture specification," *Deliverable 3.1 of IST SMS project.*
[10] A. Held, S. Buchholz, and A. Schill, "Modeling of context information for pervasive computing applications," In Proceedings of SCI 2002/ISAS 2002 (2002).
[11] T. Strang and C. Linnhoff-Popien. "A context modeling survey," *In Proceedings of the Workshop on Advanced Context Modelling, Reasoning and Management*, 2004.