



# Modular and Extendible Context Provisioning for Evolving Mobile Applications and Services

Michael KNAPPEMEYER<sup>1,2</sup>, Ralf TÖNJES<sup>1</sup>, Nigel BAKER<sup>2</sup>

<sup>1</sup>University of Applied Sciences Osnabrück, P.O. Box 1940, 49009 Osnabrück, Germany

Tel: +49 541 969 3453, Fax: +49 541 969 13453

Email: {r.toenjes, m.knappmeyer}@fh-osnabrueck.de

<sup>2</sup>Mobile and Ubiquitous Systems Group, UWE, Coldharbour Lane, Bristol BS16 1QY, UK

Tel: +44 7798742857, Fax: +44 117 3282734, Email: nigel.baker@uwe.ac.uk

**Abstract:** The provision of ambient intelligence in pervasive computing environments and smart spaces requires the evolution of context-aware systems. This paper contributes to a context provisioning framework for general purpose context-aware mobile communication systems. The presented framework allows for extendibility and modularity, hence supporting emerging and evolving services and applications. The incorporation of efficient, robust and scalable context inference is envisaged by extending context management models and in particular context distribution mechanisms. Reasoning and interpretation are crucial tasks for deriving high-level contextual information from lower level context or raw data (e.g. databases or sensor output). Context inference typically comprises several processing stages which are reflected in the elaborated framework. A subject-based provider-consumer context management model is applied. Context Providers support plug-and-play like registration functionalities and can be added or removed during runtime.

**Keywords:** Context Management, Context-Awareness, Reasoning

## 1. Introduction

Recent research in the area of mobile communication systems reveals a trend towards context-awareness and pervasiveness. Ubiquity is becoming a reality, i.e. a lot of user terminals are always interconnected by utilising infrastructure flat rate models (e.g. 2G/3G) or ad-hoc low range communication (e.g. Bluetooth). Mobile phones have already become permanent companions in everyday life and are evolving in terms of processing power, sensing capabilities, equipped memory and usability. Given these facts, offering pervasive and intelligent services, networks and environments appears to be the logic consequence on the long-term roadmap as envisioned by Weiser [5]. A system is defined to be context-aware if it is able to assist its users without need for their explicit interactivity. Instead, the system is aware of the user's context. It is able to proactively support the identified user goals and reduce disruption. Contextual information may be inferred from physical sensors (e.g. acceleration, light intensity), virtual sensors (e.g. calendar, email content) and logical sensors (e.g. databases) [2]. In the future, the number of context sources and sinks are expected to increase tremendously. At the same time new applications and usage scenarios may arise, requiring a scalable, extendible (wrt. new context types/domains) context provisioning framework and evolving context models. In particular, Social Community services (cp. Facebook, LinkedIn) are emerging. In addition, Mobile Advertising services

become popular and advertisement companies are very interested in more information about the users' context to provide tailored advertisements. According to the Financial Times Germany, sales of € 13.5 billions are expected in 2012. Specific companies invest up to € 1.5million for their mobile marketing. Thus, there is a strong demand for context-aware systems and for investigation of dynamic ad-hoc group recognition. Due to constrained resources of mobile handheld devices, complex reasoning tasks have to be performed on high performant infrastructure entities. The question is how and where to distribute knowledge and the inference process itself. Robust reasoning and inference mechanisms also need to compensate incomplete knowledge, uncertain and imprecise contextual and sensor data by incorporating context meta information, such as the degree of uncertainty or accuracy [4]. The paper contributes to designing a flexible and extendible context provisioning framework while focussing on context-aware distributed reasoning and efficient context diffusion. The framework elements are conceptualised from both a model-theoretic and an engineering perspective. The remainder of the paper is structured as follows. Section 2 defines the terminology before related work is presented in section 3. Section 4 provides an overview of the entire end-to-end context-aware system. Our framework is then presented in section 5 and section 6 finally concludes the work.

## 2. Definition and Terminology

According to Abowd et al. [6] *context* can be "any information that can be used to characterise the situation of an entity (person, place, physical or computational object) that is considered relevant to the interaction between entity and application." His definition is by far the most cited in literature and adopted in this paper. Zimmermann [7] proposes five fundamental categories of context information: time, location, activity, relations, and individuality. Regarding the definition of a *situation* in context-aware systems, different views exist in the research community. Zimmermann defines it as "the state of a context at a certain point (or region) in space at a certain point (or interval) in time, identified by a name" [7]. Being a structured representation of a part of the context it can be compared to a snapshot taken by a camera. Location and time can be used as spatio-temporal coordinates. With regard to situation-awareness Billings [8] defines a situation as "an abstraction that exists within our minds, describing phenomena that we observe in humans performing work in a rich and usually dynamic environment." In conclusion, a situation may contain an infinite variety of contextual information. In this paper a situation describes the overall state at an instant of time comprising several entities with their respective contexts. To establish a common understanding, the following terms are introduced for referring to functional components of a context-aware system:

*Context Modelling* covers

- the abstraction of the reality that is relevant for context detection,
- the identification of an adequate context representation, incl. modelling of context meta information such as context quality, and
- the provision of a context query model and mechanisms for querying.

*Context Reasoning* includes

- the interpretation that assigns a semantic to the sensor data and
- the inference that derives new conclusion about the context from existing knowledge (e.g. a priori knowledge and context history).

*Context Management* deals with

- Context Discovery as a mechanism to locate and access context sources,
- Context Acquisition as a mechanism to obtain context from diverse sources,
- Context Fusion for merging correlated contextual information, and

- Context Dissemination for efficiently propagating the context while ensuring availability and reliability.

*Context Provisioning* involves the described tasks of context modelling, context reasoning and context management as well as making it available for usage to the application or other functional infrastructure elements [3].

### 3. Related Work

The Georgia Institute of Technology elaborated a Context Toolkit offering standard libraries to programmers [12]. Reusable components were designed to assist in the development of context-aware applications. The Context Broker Architecture (CoBrA) developed by the Univ. of Maryland supports context-aware applications in smart spaces. The Context Fusion Network “SOLAR”, realised at Dartmouth College, serves as infrastructure for context acquisition. Based on a graph abstraction it provides data fusion services for aggregation and interpretation of sensor data [16]. The PACE project at the University of Queensland aims at providing a context management infrastructure. By offering functionalities for context gathering, context management and context dissemination, the development of context-aware applications is facilitated [17]. The iCAP system (University of Berkeley) allows for specification of a rule-based reaction for context-aware applications being configured by graphical user interfaces [18]. This approach is hence limited to static and well-specified rules without considering e.g. fuzzy or probabilistic logic. The successor system CAPpella focussed on providing a prototyping environment for recognition-based context-aware applications. It allows for learning user behaviour autonomously [19]. The Service-Oriented Context-Aware Middleware (SOCAM) project introduced an architecture for building and rapidly prototyping context-aware mobile services. It concentrates on a central server entitled context interpreter, acquiring context data from distributed context providers [20]. Another example for a centralised middleware approach designed for context-aware mobile applications is the project Context-Awareness Sub-Structure (CASS), allowing for further extendibility [21]. The Gaia middleware has been introduced as one of the first middlewares for supporting active smart spaces. It builds on well-known operating system design principles and adds abstraction layers for ubiquitous computing [22]. The CORTEX system, another context-aware middleware solution, is based on the Sentient Object Model. It has been designed for the development of context-aware applications in an ad-hoc mobile environment [23]. Obviously, numerous middleware solutions and context-aware systems have been proposed in the last two decades. However, none of them has concentrated on a context provisioning framework as targeted in this paper. Modularity is partly supported in existing approaches but with regard to rich extendibility, no system provides a plug and play like solution. Neither context- and knowledge-aware adaptation of the reasoning and inference process are supported, nor evolving context models and evolving applications and services.

### 4. System Architecture Overview

In the C-CAST project [1] an end-to-end context-aware content provisioning system is targeted. By inferring user and group situations we aim at selecting and providing appropriate content autonomously. The efficient distribution of situation-tailored content (video, audio, text) utilises transparent multi-party transport mechanisms, hence C-CAST investigates an innovative combination of the two competence areas *multicast/broadcast* and *context-awareness*. Figure 1 illustrates the overall system encompassing several logical layers. Trigger based adaptation of the system involves Actuator, Control and Decision Support Layer. However, this paper concentrates on context provisioning, i.e. on the Semantic Layer, the Context Acquisition & Distribution Layer and the Sensor Layer.

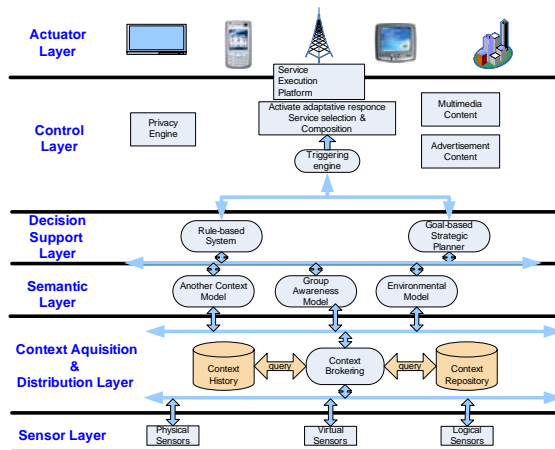


Figure 1: System Architecture

## 5. Context Provisioning Framework

In this section our concept of an extendible modular Context Provisioning Framework is introduced by covering context modelling, context management and context reasoning.

### 5.1 Context Modelling and Representation

Context models can generally be classified into six different model categories: Key-value models, markup scheme models, graphical models, object oriented models, logic based models and ontology based models [2]. Ontology design and maintenance are both challenging and tedious. Ontology parsing and reasoning come with requirement for large amounts of memory and processing power. In order to support resource constrained mobile devices and to allow for fast reasoning, we decided to build our model upon a light weight markup scheme design. We extended the Context Meta Language (ContextML) [9]. Some context meta information has been added to increase expressiveness. A context attribute has been added to reflect the state in the context lifecycle [10] and allow rapid changes between states (e.g. active, suspended). Accuracy and degree of uncertainty are introduced as optional Quality of Information parameters. Therefore, probabilistic reasoning can be supported where required. ContextML 2.0 has been designed and chosen as context representation for efficiently exchanging contextual information between network entities. However, the Context Provisioning Framework is not restricted to this context model at all. Inside the context providing entities other context models may be utilised as explained later in more detail. An exemplar snippet of ContextML can be found below in Figure 2.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<contextML xmlns="http://ContextML/2.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ContextML/2.0 http://127.0.0.1:8070/schemas/ContextML-2.0.xsd">
<ctxEls><ctxEl>
  <contextProvider id="LCP" v="2.00"/>
  <entity id="michael" type="username"/>
  <scope>position</scope>
  <metaPart>
    <timestamp>2009-01-14T18:03:29+01:00</timestamp>
    <expires>2009-01-14T18:33:29+01:00</expires>
    <state>active</state>
    <QoI>
      <accuracy>580</accuracy>
      <probability>0.9</probability>
    </QoI>
  </metaPart>
  <dataPart>
    <par n="latitude">22.73</par>
    <par n="longitude">10.4</par>
  </dataPart>
</ctxEl></ctxEls>
</contextML>
```

Figure 2: ContextML Snippet

## 5.2 Context Management

A well-known and often used speciality of the combined context server and networked service model is the producer-consumer role model. Network components may take the roles of a Context Provider (CP) or of sinks, i.e. Context Consumers (CC) [10][13]. These basic two role entities are interconnect by a Context Broker (CB) providing a directory and lookup service (cp. Figure 3). Registration and lookup of Context Providers is based on subjects of interests, i.e. context scopes/type (e.g. geographic location) and the entities (e.g. user, terminal) related to the contextual information. The CPs constitute the spine of the framework. A CP provides new system knowledge and contextual information to the system. As illustrated in Figure 4 a CP may reason about internal data sources (e.g. databases, raw sensor data output). Alternatively, the CP may depend on contextual information provided by other external CP. Hence, various CP may act as CC for deriving high-level situational context from primitive context. The Context Reasoner logic is tailored to the desired output context information as further discussed in section 5.3. A simplified overview of the entire Context Provisioning Framework design is depicted below in Figure 5. Context-aware applications and services can use defined Service Enablers to access contextual information and define event triggers. Those Service Enablers are placed at the Service Enabler plane and hide functionalities by mediating to the actual context provisioning framework. To reduce the concern of a single point of failure and to allow for large scale systems, the concept of CB federations has been introduced. Several CB may collaborate by exchanging registration and lookup data. To ensure robustness and increase performance, a context cache is attached to the CB. The history database may store obsolete context information required for learning and inference from previous events. The framework contains various CP. Each of them implements ContextML parsing and performs an individual reasoning. The CC can easily acquire context information by invoking an on-demand request. In addition, a CP may optionally allow for subscribing to defined events (e.g. threshold exceedance, context change, timeout). The CC is then notified in case of event occurrence. Both pull and push communication applies a Representational State Transfer (REST) [14] interface on top of the Hypertext Transfer Protocol (HTTP). Consequently, both control and data interface utilise defined Uniform Resource Identifiers (URI) to provide access to internal CP and CB methods. Both data replies and control reply messages (e.g. ACK, NACK) are encoded in ContextML and transferred via HTTP. For underlining the simplicity some examples are given:

- `http://server.de:8080/CB/providerAdvertising?<contextMLadvertisement>`
- `http://server.de:8080/CB/getContextProviders?scope=position`
- `http://server2.de:8080/LCP/getContext?entity=michael&scope=position`
- `http://server3.de:8080/LCP/ContextBroker/subscribe?entity=michael&scope=position&event=change&callbackUri=<...>`

As illustrated in Figure 5 the framework includes various Context Providers. The Calendar CP provides access to calendar entries by parsing iCalendar (RFC 2445) files and maintaining an internal database. The Location & Proximity CP utilises reported GPS coordinates and cell IDs in order to determine the entity's outdoor locations and proximity

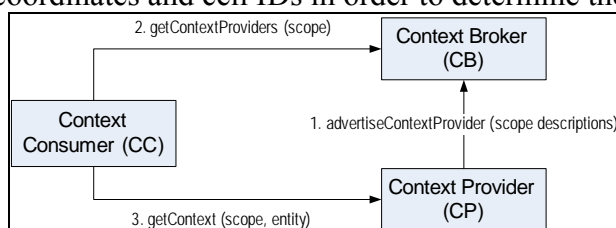


Figure 3: Context management role model

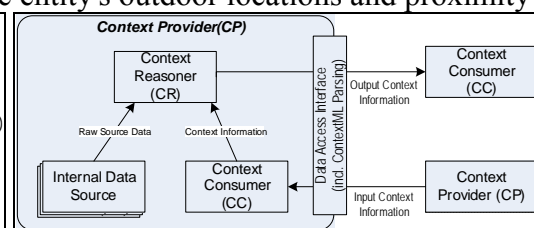


Figure 4: Context provider logical design

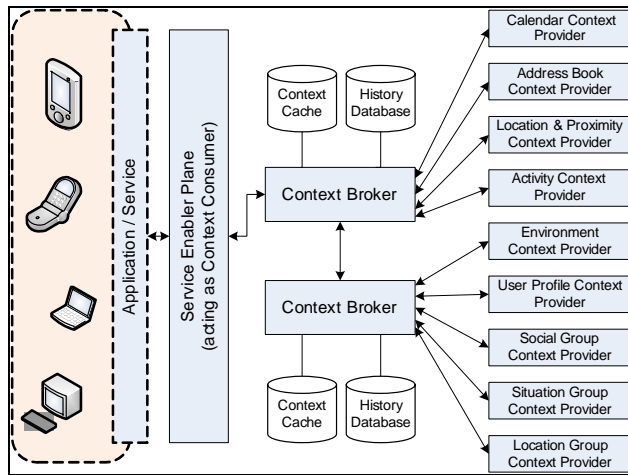


Figure 5: Context provisioning framework

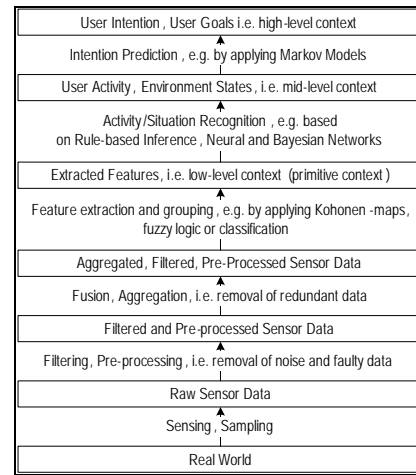


Figure 6: Detection processing stages

to other entities. Currently an additional feature is being elaborated for determining indoor location and logical locations, e.g. building, street name or city name. The User Profile CP provides rather static data such as the date of birth, gender and occupation. Moreover, self-defined or inferred interest profiles may be stored. The Activity CP depends on the described low-level context and reasons about user activities or situations. Based on calendar, location and proximity data, this CP is able to determine whether a user is participating in a formal meeting with colleagues or sitting at home and having guests. The current implementation relies on rule-based reasoning [15] but faster and more scalable solutions are in progress. Since the research project C-CAST aims at providing context-aware multiparty services, Context Providers for dynamic group recognition are of particular interest. Such groups can be estimated based on numerous criteria, e.g. common interests or common relations (Social Group CP), common situations (Situation Group CP) or physical proximity (Location Group CP).

### 5.3 Context Reasoning

Reasoning is the process of deriving abstract high-level context information (e.g. user is in meeting) from low-level primitive context (e.g. GPS longitude, light intensity). Figure 6 depicts processing stages and context abstraction levels. Some of the reasoning mechanisms are borrowed from the areas of artificial intelligence and knowledge-based systems. With regard to our context management framework, reasoning is not only distributed amongst abstraction levels but also amongst numerous network entities. Each Context Reasoner (CR; cp. Figure 4) inside the Context Providers contributes to the process. A CR may serve different layers of the context inference chain. An exemplar Environment CP or Motion Detection CP may implement lower abstraction layers by processing (sensing, filter, fusion) raw physical sensor data. This extracted primitive context is in turn required by an Activity CP for reasoning about user activities. This example highlights the close relationship between Context Provider entities. It becomes obvious, that efficient and scalable context diffusion mechanisms are essential. The supported publish-subscribe triggering reduces communication overhead and allows for propagating context changes rapidly. The reasoning has to deal with uncertainty and imprecision. A proposition is uncertain if it can not clearly be defined as true or false (e.g., the temperature of 21 degrees Celsius suggests an indoor location with a probability of 80 percent). A proposition is imprecise if it has no accurate value but a range of several values (e.g., a warm climate is between 21 and 25 degrees). Several schemes have been suggested to represent and combine uncertainty, such as possibility theory, Bayes nets, and evidence theory, and to model imprecision by fuzzy

sets or linguistic variables. The presented approach employs possibility theory [24] to model uncertainty and fuzzy sets to model imprecision.

**Modelling Uncertainty:** To judge the uncertainty of a proposition, a measure of belief is defined. It maps all propositions into the interval [0, 1]. A hypothesis that has not yet been tested in the sensor data is neither necessary right nor wrong. To model this ignorance the interval [0, 1] is divided into the three intervals necessity:  $N(e)$ , necessity of the contrary proposition  $N(\neg e)$ , and ignorance  $\Theta(e)$  (cp. Figure 7). If no knowledge about a proposition  $e$  exists, both, necessity  $N(e)$  and necessity of the contrary proposition  $N(\neg e)$  are zero (The necessity measure in the sense of Dubois and Prade [24] assumes that the elementary focal propositions can be ordered in a hierarchy of inclusion, i.e., are consonant). The possibility  $P(e)$  of a proposition is given by  $P(e) = 1 - N(\neg e)$ . The comparison of proposition, i.e. hypothesis, with the sensor data, i.e. evidence, reduces the uncertainty by increasing the necessity of  $e$  or its opposite  $\neg e$ .

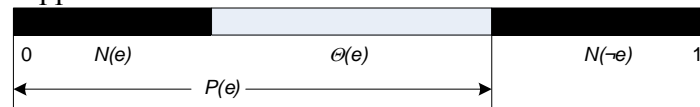


Figure 7: Necessity  $N(e)$  and possibility  $P(e)$

**Modelling of Imprecision:** To model the imprecision of a proposition, fuzzy sets in the sense of Zadeh [25] are employed. They describe the membership of a value  $x$  to a set, e.g. hypothesis  $H$ , with a membership function in the interval [0, 1] (cp. Figure 7). A certain membership value  $e$  is interpreted as possibility  $P(e)$  that a proposition “ $x$  possesses the value  $e$ ” (e.g., the temperature is 21 degrees Celsius) is true for the assumption “ $x$  is  $H$ ” (e.g., the location is indoor).

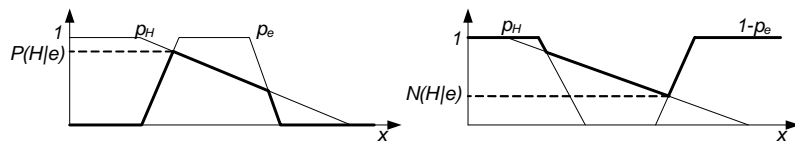


Figure 8: Computation of possibility  $P$  and necessity  $N$  of hypothesis  $H$  for a given evidence  $e$

The combination rules that are defined for fuzzy sets allow the evaluation of imprecise attributes. The possibility and necessity of an imprecise hypothesis  $H$  for a given imprecise measurement  $e$  are depicted in Figure 8 and are determined according to following equations:

$$P(H | e) = \sup_{x \in X} \min (P_H(x), P_e(x)); N(H | e) = \inf_{x \in X} \max (P_H(x), 1 - P_e(x))$$

## 6. Conclusion and Future Work

In this paper we presented a modular general purpose Context Provisioning Framework. Due to the chosen consumer-producer model, it is extendible during run-time by adding new Context Providers. Thus, evolving applications and services can be supported by introducing new context dimensions, i.e. context scopes. For exchanging contextual information and control messages between context processing entities, the meta language ContextML has been extended. Based on the described framework, we plan to specify description languages for CP interfaces and capabilities. Currently, the CP advertisement only includes context scopes and access URIs. The framework would benefit from CP information about provided events, supported communication paradigms (publish/subscribe vs. on-demand request), supported context query models and other functionalities. These would enhance the plug and play behaviour significantly. In addition, the event management is further being investigated, covering suited event modelling and efficient propagation.

## Acknowledgment

This work has been supported by the European FP7 ICT project C-CAST [1] which aims at evolving mobile multimedia multicasting to exploit the increasing integration of mobile devices with our everyday physical world and environment. .

## References

- [1] ICT FP7 Research Project Context Casting (C-CAST), web site: <http://www.ict-ccast.eu>
- [2] Baldauf, M., Dustdar, S. & Rosenberg, F., "A survey on context-aware systems", *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4), 2007.
- [3] Riva, O. & di Flora, C., "Contory: A Smart Phone Middleware Supporting Multiple Context Provisioning Strategies", 26th IEEE International Conference on Distributed Computing Systems Workshops, 2006.
- [4] Anagnostopoulos, C. & Hadjiefthymiades, S., "Enhancing Situation-Aware Systems through Imprecise Reasoning", *IEEE Transactions on Mobile Computing*, 2008.
- [5] Weiser, M., "The computer for the 21st century", *Human-computer interaction: toward the year 2000*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995.
- [6] Abowd, G.D. et al., "Towards a Better Understanding of Context and Context-Awareness", *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, 1999.
- [7] Zimmermann, A., "Context Management and Personalisation", PhD Thesis, University of Aachen, 2007.
- [8] Billings, C., "Situation Awareness Measurements and Analysis: A Commentary", *Proc. Int'l Conf. Experimental Analysis and Measurement of Situation Awareness*, 1995.
- [9] Moltchanov, B. et al., "Context-Aware Content Sharing and Casting", *ICIN 2008, Bordeaux*, 2008.
- [10] Hyunjun Chang, Seokkyoo Shin & Changshin Chung, "Context Life Cycle Management Scheme in Ubiquitous Computing Environments", *International Conference on Mobile Data Management*, pp. 315-319, 2007.
- [11] Strang, T. & Linnhoff-Popien, C., "A Context Modeling Survey", *The Sixth International Conference on Ubiquitous Computing, Workshop on Advanced Context Modelling, Reasoning and Management*, Nottingham/England, 2004
- [12] Dey, A., Salber, D. & Abowd, G., "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications", 2001
- [13] Ramparany, F. et al., "An open context information management infrastructure the IST-amigo project", *3rd IET International Conference on Intelligent Environments*, 2007.
- [14] Fielding, R. T., "Architectural Styles and the Design of Network-based Software Architectures", PhD Thesis, University of California, 2000.
- [15] Goix, L. et al., "Situation Inference for Mobile Users: A Rule Based Approach", *International Conference on Mobile Data Management*, 2007. pp. 299-303.
- [16] Chen, H. et al., "Intelligent agents meet semantic web in a smart meeting room", *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, 2004, pp. 854-861.
- [17] Henriksen, K. et al., "Middleware for Distributed Context-Aware Systems", *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*. pp. 846-863.
- [18] Sohn, T. & Dey, A., "iCAP: an informal tool for interactive prototyping of context-aware applications", *CHI '03 extended abstracts on Human factors in computing systems*. Ft. Lauderdale, Florida, USA: ACM, 1003, pp. 974-975.
- [19] Dey, A. et al., "A CAPPella: Programming by demonstration of context-aware application", 2004.
- [20] Tao Gu, Hung Keng Pung & Da Qing Zhang, "A middleware for building context-aware mobile services", *IEEE 59th Vehicular Technology Conference*, 2004, pp. 2656-2660.
- [21] Fahy, P. and Clarke, S., "CASS – a middleware for mobile context-aware applications", *Workshop on Context Awareness, MobiSys 2004*.
- [22] Roman, M. et al., "A middleware infrastructure for active spaces", *IEEE Pervasive Computing*, 2002, 1(4), 74-83.
- [23] Biegel, G. & Cahill, V., "A framework for developing mobile, context-aware applications", *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications*, 2004, pp. 361-365.
- [24] Dubois, D., and H. Prade, "Possibility Theory: An Approach to Computerized Processing of Uncertainty", *Plenum Press*, New York and London, p 263 ff, 1988.
- [25] Zadeh, L. A., "A Theory of Approximate Reasoning", *Machine Intelligence*, vol. 9 (J. Hayes, et al., editors), Halstead Press, New York, pp. 367 – 412, 1979.