Applying Semantic Technologies for Context-Aware AAL Services: What we can learn from SOPRANO

Peter Wolf
FZI Forschungszentrum Informatik, Karlsruhe, Germany
wolf@fzi.de

Andreas Schmidt
FZI Forschungszentrum Informatik, Karlsruhe, Germany
aschmidt@fzi.de

Michael Klein
CAS Sofware AG, Karlsruhe, Germany
Michael.Klein@cas.de

Abstract: Ambient assisted living (AAL) is a newly emerging term describing a research area with focus on services that support people in their daily life with particular focus on elderly people. This includes reminding and alerting the assisted person(s) and their environment, giving feedback, advice, and impulses for physical, or social activities, among others. All of these supportive actions need to be context-aware. Semantic technologies have been considered to be a perfect fit for context-awareness in pervasive computing and ambient intelligence. Mainstream semantic (web) technologies are nowadays largely based on description logics and the W3C standard OWL-DL (c.f. [WZGP04], [DB08], [PVdBW$^+$04], and [ESB07]), which are also used in the AAL domain (see [PG07] and [KC06]). The analysis of the scenarios and use cases, however, have yielded requirements and constraints, which have shown that these mainstream technologies are not well-suited for the AAL domain. In this paper we present the SOPRANO approach as an alternative semantic approach to capturing, managing, and enriching context information for context-aware AAL services.

1 Introduction

Ambient assisted living (AAL) is a newly emerging term describing a research area with focus on services that support people in their daily life with particular focus on elderly people. This includes reminding and alerting the assisted person(s), giving feedback, advice, and impulses for physical, or social activities, among others. All of these supportive actions need to be context-aware, i.e., they require a rather deep knowledge about the situation of the assisted person, including location, activity, vital status, environmental conditions, social network and quality of social relationships etc.

Semantic technologies have been considered to be a perfect fit for context-awareness in pervasive computing and ambient intelligence (e.g. [WZGP04], [DB08], [PVdBW$^+$04], and [ESB07]). They provide ontology formalisms for representing complex domains and

complex system behaviour in those domains that are machine processable and on a sufficient level of abstraction, so that they are easy to maintain, extend, and to be reused for different scenarios. Mainstream semantic (web) technologies are nowadays largely based on description logics and the W3C standard OWL-DL[1], which are also used in the AAL domain (see [PG07] and [KC06]).

Within the Integrated Project SOPRANO, we have embarked on an ontology-based approach for a semantic middleware for context-aware AAL services as part of an ontology-centred design methodology (see [KSL07]), bringing together all stakeholders in the system development with the help of an evolving ontology as a mediating artefact. The analysis of the scenarios and use cases, however, has yielded requirements and constraints, which have shown that these mainstream technologies are not well-suited for the AAL domain.

In this paper we present the SOPRANO approach to capturing, managing, and enriching context information for context-aware AAL services that is based on a dedicated context management component. The next section summarises typical requirements derived from the AAL domain. Section 3 compares the SOPRANO approach to other context managing solutions that put a focus on semantic web standards. Section 4 elaborates on the main concepts of the SOPRANO Context Manager (CM) component and their relation to the requirements presented in Section 2. The last section concludes the document with summary and outlook.

2 Specific Requirements for Context Management in AAL

SOPRANO aims at delivering context-aware services to older people in their own homes. Thus, these services should be able to incorporate knowledge about the current situation of the assisted person (AP). This would allow services to tailor their specific functionality to the existing needs and requirements. An example of this context information could be the fact that the AP is watching-TV or that the AP is hearing-impaired. This information would trigger a TV-based service, scheduled to remind about certain medication, to inform the person via an onscreen message on the TV, instead playing a sound message. This kind of context-awareness requires the service to access a model of the AP's current situation involving notions like activities and impairments. This functionality is typically provided by a context management component. In the following, the requirements regarding this context manager are described from the sensor's (Section 2.1) and service's (Section 2.2) point of view.

[1] Web Ontology Language, website: http://www.w3.org/TR/owl-features/

2.1 Requirements from the sensors' perspective

In order to be able to uniformly access sensors, every sensor that provides information about the AP's environment should be seen as a **sensor service**. This includes software modules that directly represent what is commonly referred to as sensor such as door sensors or electricity-consumption detectors but also more complex applications like speech-recognition services.

Based on this definition, sensor information exhibits certain problems:

- Information delivered by sensors is often **uncertain and erroneous** (see [HI04] for an in-depth analysis), as for example speech recognition services (due to noise or unclear pronunciation). The system should be able to cope with that fact.
- Furthermore, a set of sensors can provide **conflicting information**. Although each sensor usually deals with contradictions in its own data, but it cannot do so with conflicts that emerge between information of different sensors (e.g., a movement sensor detects the AP in the living room, which the RFID scanner detecs the AP at the entrance door). A context management component must be able to deal with issues in modelling and reasoning and hide the complexity from the services using the context information.
- Additionally, the context manager should enable the **independent contribution** of sensor technology providers. This requires for a sufficiently easy (and decoupled) interface together with a semantic model that is tailored towards simple sensor integration. This ensures that low-cost (because they do not need to perform complex abstractions) and universal-purpose components can be plugged into the system with minimum adaptations. E.g., an RFID should be easy to plugin in without the scanner knowing about the carrier of the tag and the meaning of the detection of a tag.

2.2 Requirements of the Services

Applications which directly provide services to the assisted person put different requirements to an ontology-based context manager. Usually, they are intended to operate on more meaningful information centering on the AP, activities, emergencies, location in symbolic representation (i.e. rooms), impairments etc. This high-level context-information partly needs to be derived from various sensor information which requires a variety of **complex algorithms**:

- Typically, these algorithms do not only react on current information, but also the history, e.g., for detecting changes in habits in daily activities. This asynchrony of acquisition and usage (see [Sch06]) makes incorporation of a **persistence storage** inevitable.
- Furthermore unlike sensors, most context-aware services deal most easily with a **conflict-free representation** of context information which requires incorporation

of conflict resolving algorithms. It would introduce a high level of complexity, e.g., to deal with multiple potential locations of a person. Such a conflict resolution has to deal with variable sensor configurations that could exhibit different forms of imperfection, and the requirements of services can be different, e.g., depending on how sensitive to errors their effect in the AP's world are. Therefore, a conflict resolution should be based on a set of configurable heuristics exploiting recency, confidence, and validity of context information.

- Additionally, a model capturing the states and environment of the AP should be **hardware-independent** to enable reuse of context-aware services in different homes with varying sensor layouts, e.g., allow for locating the AP with movement sensors as well as RFID or even cutting-edge radar technologies.

In the following section we show that current context management components are not able to cope with the specific requirements of AAL systems.

3 Related Work

Many approaches to context management are based on semantic technologies especially ontologies. Ontologies capture a shared conceptualization of context defined within a formal explicit specification. They ease development of the context management component as well as context-aware services.

Some systems go even further and employ web standards as e.g. the knowledge representation language OWL-DL. OWL is suitable for open and distributed nature of the web but, as we argue, is not suitable for capturing problems emerging in intelligent local environments. More specifically, it yields the following problems, which are due to its description logics foundations:

- Uncertainty is not incorporated into OWL-DL reasoning
- Reasoning on conflict-prone data will lead to unanticipated results or unsatisfiability
- Reasoning on temporal data, which is needed to exploit historical data, is hard to achieve in OWL and usually highly inefficient
- Standard OWL-reasoning even incorporating rule languages like SWRL is not sufficient for applying context-reasoning which is able to detect more complex context-information like activities, emergencies etc. Rather, the complexity of problems in context reasoning suggests that we should use a formalism-neutral context management component that allows for combining multiple strategies, e.g., data mining approaches for activity pattern detection, Bayesian networks for combining multiple and uncertain sensor input, as well as rule-based formalisms for more descriptive specification (derived from analytical knowledge)

- Semantics of cardinality constraints and other restrictive features (needed for checking sensor conformance and for specifying a contract towards the users of user context management system) cannot be expressed by OWL-DL. Instead, these constraints should be enforced by processing conflicted data by means of conflict resolution heuristics (as outlined in Section 2).

Furthermore, many OWL reasoning features are not useful in such local systems:

- Context information is distributed, but collected within one component
- Is-a relations only have to be exploited in simple queries
- Usually, abstraction of sensor information is not naturally to be translated into a classification problem
- Only a small subset of modelling constructs as provided by OWL-DL needs to be exploited

These problems can be seen, for example, in the approach of Gu et al. (see [XWPZ04]). Gu et al. proposed a high-level context model for intelligent environments called the CONON ontology. Upon this model they have built a context managing component called service-oriented, context-aware middleware (SOCAM). SOCAM enables a process of semantic abstraction from sensor information into CONON compliant information (see [XWPZ04] and [WZGP04]). Since they rely on OWL-reasoning in conjunction with inference rules it is arguable whether detection of complex activities or emergencies is possible, e.g., the gradual change in certain habits. Additionally, the process of semantic abstraction and the handling of conflicted data is not clearly outlined.

Similar considerations apply to the work of Ejigu et al. (see [ESB07]) who present an ontology-based generic context management (GCoM) model. Although they provide a model on different levels of abstraction they solely rely on OWL-reasoning and compatible rule languages.

The DogOnt model (see [DB08]) provides a complete conceptualization of sensor, device, and network functionalities of domotic environments. Domotic House Gateway of Pellegrino et al. which is based on DogOnt aims at providing automatic device cooperation which can be dynamically and automatically updated to accommodate necessities and anticipate users' actions (see [PBC06]). Unlike SOPRANO the OWL-based DogOnt model captures functionality on a very low-level only. A semantic abstraction is not incorporated. Furthermore, problems of uncertain or conflicted information are not explicitly addressed.

SOPRANO's approach to context management, which we will describe in the following section, is tailored to the requirements as described in section 2. The SOPRANO Context Manager does therefore provide two layers of abstraction tailored to sensor and service providers, respectively. These two layers are connected by a set of context reasoning algorithms converting sensor states into AP states. This two-layered approach can deal with conflicting sensor information and provides a conflict-free view for context-aware services.

4 SOPRANO Context Manager: Concepts and Implementation

The main concept of the SOPRANO Context Manager (CM) has been derived from the work done by Schmidt (see [Sch06]) leading to a system that can handle conflicts and uncertainty and is able to derive a conflict-free view on context information. Within SOPRANO, these concepts have been merged with the two semantic layers as sketched above (see Fig. 1).

To comply with requirements in section 2, SOPRANO provides a two-layered semantic representation of the environment: the sensor-level ontology providing a description of sensors and sensor states and the AP-context ontology describing the situation of the AP in terms of activities, symbolic location, emergencies, impairments, and devices. Additionally, a two-layered approach in terms of conflict resolution is used in SOPRANO: with a typically conflict-prone knowledge sink at the bottom where all the various sensors can dump their data to without losing or rejecting any information and a conflict-free upper layer that can easily be exploited for context-awareness.

The whole SOPRANO system is based on the Open Service Gate Initiative[2] (OSGi) framework. OSGi is an object and component middleware that enables easy provision and usage of services together with means for synchronous and asynchronous messaging. The SOPRANO Context Manager provides several OSGi-based interfaces tailored to sensors and services by means of distinct semantic models. These interfaces are described in detail in the following sections.

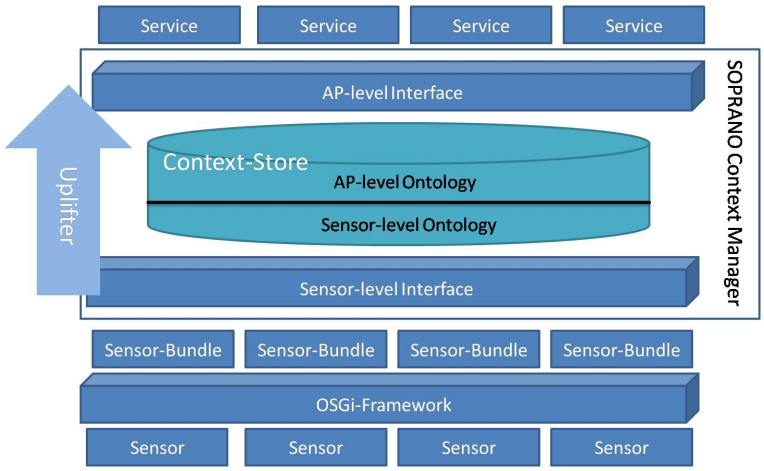

Figure 1: SOPRANO Context Manager Architecture

[2]OSGi-Website: http://www.osgi.org/Main/HomePage

4.1 The Sensor-level Interface

SOPRANO does provide an interface accessible through the OSGi event-admin that allows sensors to push their current states into the CM. Sensors that want to push their data into the CM have to comply with certain syntactic and semantic contracts (see [WSK08]). The mappings between sensor-internal communication semantics and SOPRANO semantics can be achieved within the bundle that connects the sensor to SOPRANO (see bundles in Fig. 1).

On a syntactic level all SOPRANO compliant sensors have to be integrated as OSGi components. Since only simple data push is supported, conversion from data pull to push paradigms has to be realized in the sensor integration bundles as well. Additionally, the sensor interface ignores continuous information streams, but instead requires sensors to commit only changed sensor states. This decreases the workload of CM. Aging of context information can account for certainty changes through time.

4.2 The Sensor-level Ontology

The information that sensors provide, must conform to the sensor-level ontology (see Fig. 2 for an extract). The sensor-level ontology models sensors and the states that they can measure.

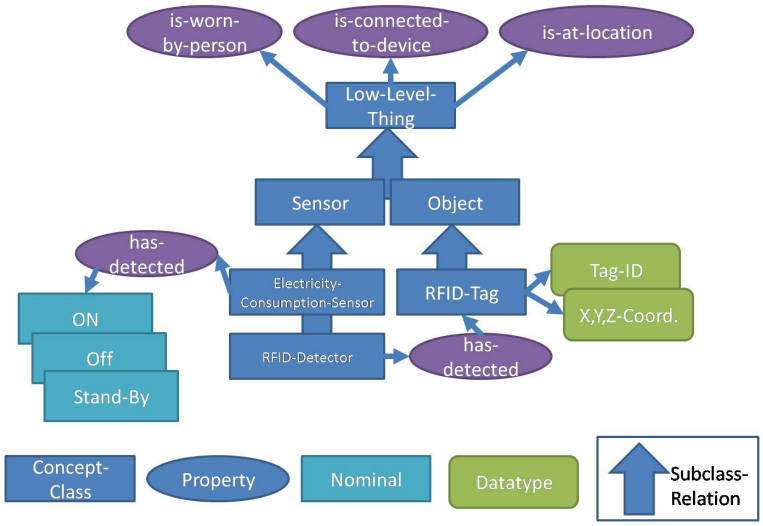

Figure 2: Extract of sensor-level ontology

Concepts: The sensor-level ontology enables easy integration of sensors by providing a sensor specific vocabulary for data-exchange. This enables sensor-technology providers to integrate the respective sensor component independently, i.e. in order to integrate a sensor, there is no cooperation with other sensor providers needed. This independent contribution is enabled by modelling individual sensors and sensor-states. Additionally, this kind of modelling enables sensor integration with substantially less effort, since the sensor-level ontology models sensors directly and leaves the transformation to more abstract terms to the CM. Additional meta-information can be attached to every piece of information in order to capture confidence, transaction time, and aging of context information.

Semantics: Sensors can transmit their information in form of context statements (extended RDF triples) which incorporate meta-information that can be exploited by augmentation and context-aware services. The following definition has been derived from Schmidt.

A *context statement* is a tuple (s, p, o, t0, v, a0, s, i), where

- s is an URI for the subject of a statement,
- p is an URI corresponding to the predicate of a statement,
- o is an URI given as the object of a statement or an atomic datatype value,
- t0 refers to the point in time, when the fact was written into the DB (transaction time),
- v is used to specify a validity duration of the context fact,
- a0 stands for the initial confidence of the statement at time t0 in percent,
- s indentifies the contributor of the context tuple, and
- i is an URI for identifying the tuple itself.

The validity of statements is further restricted by relation to a given conceptual model. Following Schmidt's database-like semantics, each instance URI must be explicitly classified beforehand, i.e. instance URI and instance-class relationship are already known to the system. Additionally, every predicate has to conform to a property of the sensor-level ontology (see Fig. 2) and subject-class and object-class have to conform to domain and range of that property.

Therefore, we can distinguish the following entities: conceptual/datatype classes, concept/datatype properties, conceptual instances, and data type values. Additionally, we introduced nominal concepts as exemplified below.

Example: Fig. 2 shows the high-level structure of what the sensor-level ontology looks like. Low-Level-Thing captures sensors and objects necessary for sensor functionality like RFID-tags. Instances of this concept can be connected to the AP-context level, like a

tag worn by a person or an electricity consumption detector that is connected to a certain device (like a TV). All modelled sensors are directly linked to states that they can detect. Sensor states can either be standard data types like double or a set of distinctive states which are then modelled in a nominal fashion. More complex detection-states like x,y,z-coordinates are subsumed under one common state which is not modelled in a nominal fashion.

4.3 Interfaces for Context-aware Services

Situational-awareness requires context-information on a more meaningful level, e.g. vocabulary describing the APs position at a certain room instead of RFID-tags within x,y,z-coordinates or status of doors, windows, oven, and fridge instead of door/window or electricity consumption sensors.

The service interface allows services to easily query context information and evaluate context conditions. Both functionalities are based on a conflict-free representation of context data. Additionally, the interface enables proactive behaviour of services (automatically reacting to certain context changes without being explicitly triggered through user involvement) by means of asynchronous messaging where service can register to the changes of certain AP-states.

4.4 The AP-level Ontology

The service interface operates on the AP-level ontology. The ontology vocabulary includes terms like activities, emergencies, location in symbolic representation (i.e. rooms), impairments, person, AP etc. Some of these AP-states need to be derived from sensor data by means of complex algorithms. Fig. 3 shows the main structure of this model based on concepts of Person, Location, and High-Level-Thing.

Concept: The AP-context ontology is centred on concepts of person, high-level-thing, and location. These concepts are interrelated with each other and connectable to particular sensors (see properties connected to Low-Level-Thing in Fig. 2). These three main classes are further subdivided into more specific classes as exemplified in Fig. 3 on the person class. All these entities can take specific states that can be subsumed under the terms person state, location-state, and high-level-thing-state.

Properties are divided into two categories: functional and non-functional properties. Functional properties restrict the set of valid statements considering a certain point in time. This means for a certain subject and predicate there can only be one valid object at a certain point in time. This can be used for modelling that a person can only be in one room at a specific point in time. The context statement definition is adopted form the sensor-level ontology.

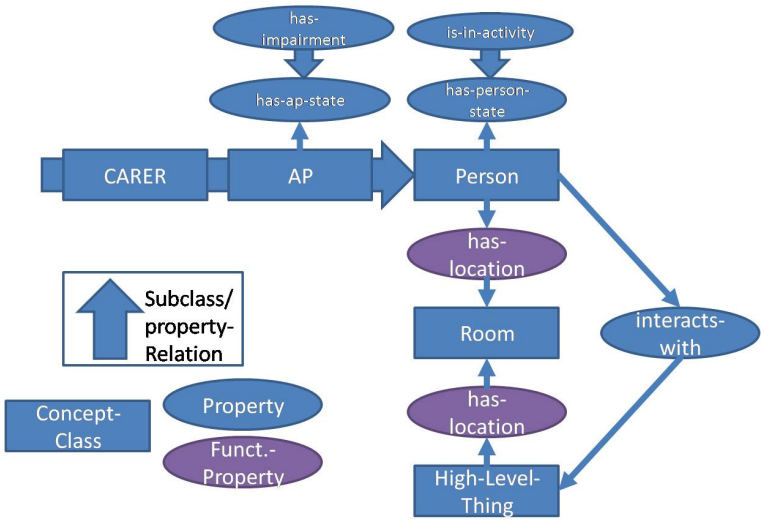

Figure 3: Extract of AP-context ontology

Semantics: In general the semantics of the sensor-level ontology have been adopted. In addition, a hierarchy on properties and classes can be defined in order to support different levels of abstraction. This hierarchy is exploited by means of query semantics which delivers more specialized statements as answers to abstract queries. Such queries are stated by means of context patterns.

Definition: A *context pattern* is a tuple (s', p', o', t0', v', a0', s'), where

- s' is an instance or class URI which is matched against the subject of a statement,
- p' is an URI corresponding to the predicate of a statement,
- o' is an instance URI, class URI, datatype, or datatype value which is matched against the subject of a statement,
- t0' refers to a date, no statement is allowed that has a t0 value corresponding to a newer date,
- v' can be true or false, where selecting true would return valid statements only (i.e. validity duration plus transaction time of a statement must result in a point in time equal to or greater then the current time),
- a0' is the minimal allowed confidence of the statement,
- s' a list that contains possible contributers

A context pattern is considered to be a simple context query, i.e one can use a context pattern to define a set of context statements and the Context Manager returns all statements

that belong to this set. Additionally, a context query will exploit the ontology class and property hierarchy. To query for the current location of the AP with minimum confidence 0.9, we can use the following context pattern (AP,is-at-location,-,now,true,0.9,-).

The service interface gives access to a conflicted view on context-information where information as written by sensors and augmentation services can be directly accessed. Additionally, the system provides a conflict-resolved view by constraining the set of statements. This is achieved by applying conflict-resolution heuristics.

A generic constraint states that only one state with a particular subject, predicate, and object can exist. Such statements could enter the context store in case two sensors are detecting the same sensor state. A specialized conflict-resolution method is applied to functional properties. The rule states that statements containing a predicate modelled as functional property connect a subject instance at most to one object-instance, i.e. the subject cannot connect to more than one object. A query upon such functional properties is also subject to conflict-resolution.

Right now we are experimenting with several conflict resolution strategies that can filter according to actuality, validity, and confidence of statements. Within the current SOPRANO implementation we are applying a strategy that returns the most recent statement, but since Schmidt (cf. [Sch06]) experienced good results with a maximum confidence strategy in combination with an aging mechanism, further tests have to be applied.

In addition to queries, services can evaluate context conditions, i.e. the CM checks whether certain statements appear in or are supported by the context database. For non-functional properties the CM returns true (statement is stored) and unknown (statement not stored). For functional properties true, unknown, and false (different object than in conditions) can be returned.

Example: According to the medication example mentioned above a display service reacts when the medication reminder is triggered and queries the context manager for impairments of the AP. According to the results a suitable modality can be chosen. Based on the condition that the AP is watching TV, the appropriate output device (e.g. TV or mobile phone) can be chosen.

4.5 Semantic Uplifting

Augmentation services provide mapping of information from sensor-level to AP-context ontology. Since this is considered to be a mapping from simple sensor-states to a more meaningful description of the AP's context we call this special kind of context-reasoning semantic *uplifting* and the algorithms that provide this functionality: *uplifter*.

The AP-level ontology is considered to be independent of the underlying sensors. Due to this fact we have developed an adaptable framework of algorithms in which uplifters can easily be added and removed. This enables us to map different sensory layouts to the same AP-level concepts. A more in-depth analysis can be found in [KW09].

Uplifters are not an integral part of the CM but can be registered via the blackboard approach. This mechanism allows for independent integration of different algorithms that are needed for activity and emergency detection. Within SOPRANO we have incorporated simple rule mechanisms as well as Hidden Markow Models and Neuronal Networks (cf. [KW09]). This approach enables uplifters to exploit internal background knowledge as stored, for example, in pre-trained HMMs but also the information that is stored within the context store, especially the relations that connect sensor-level elements to AP-level elements.

5 Summary, Conclusion, and Outlook

In this paper we presented the SOPRANO approach to context management for the AAL domain. We have argued that semantic web standards like OWL-DL are not suited for the problems encountered in the AAL domain. This is mainly due to the importance of flexible handling of uncertainty (as sensors provide information that is not 100 percent reliable), conflicts (as use of multiple sensors may result in different information on the same real-world fact), and history (as we need to analyze the history to detect deviations from usual behavior and activity patterns) and the non-appropriateness of the reasoning capabilities of OWL (e.g., classification problems). Rather, the complexity of problems in context processing (conflict resolution strategies, semantic uplifting) suggests that we should use a formalism-neutral context management component that allows for combining multiple strategies, e.g., data mining approaches for activity pattern detection, Bayesian networks for combining multiple and uncertain sensor input, as well as rule-based formalisms for more descriptive specification (derived from analytical knowledge).

As a response, we propose the SOPRANO approach to context management for the AAL domain. Our approach is based on two semantic representations that capture lower-level sensor states as well as more meaningful AP-context information. Complex algorithms that can exploit context data and background knowledge can provide a mapping between these two knowledge bases. Furthermore, this approach, being based on the work done by Schmidt (see [Sch06]), allows for a context fact storage where potentially conflicting sensor data is collected. Context-aware services are provided with a conflict-free representation that is derived by applying configurable conflict resolution heuristics. Also, the model explicitly addresses uncertain and error-prone sensor information by capturing the uncertainty and allowing for specifying the acceptable level of uncertainty.

The SOPRANO Context Manager has already been used to operate a living lab at FZI with diverse sensors and actuators, and will be evaluated with as part of the SOPRANO Ambient Middleware in large field trials at the end of 2009 in real homes.

As part of the SOPRANO Ambient Middleware, the context manager component will be released as open source (see http://openaal.org).

References

[DB08] Fulvio Corno Dario Bonino. DogOnt - Ontology Modeling for Intelligent Domotic Environments. In *International Semantic Web Conference*, pages 790–803, 2008.

[ESB07] Dejene Ejigu, Marian Scuturici, and Lionel Brunie. An Ontology-Based Approach to Context Modeling and Reasoning in Pervasive Computing. pages 14–19, March 2007.

[HI04] Karen Henricksen and Jadwiga Indulska. Modelling and Using Imperfect Context Information. *Pervasive Computing and Communications Workshops, IEEE International Conference on*, 0:33, 2004.

[KC06] Eunhoe Kim and Jaeyoung Choi. An Ontology-Based Context Model in a Smart Home. In *Workshop on Ubiquitous Web Systems and Intelligence (UWSI 2006)*, pages 11–20. pringer Berlin / Heidelberg, 2006.

[KSL07] M Klein, A. Schmidt, and R. Lauer. Ontology-Centred Design of an Ambient Middleware for Assisted Living: The Case of SOPRANO. In *Towards Ambient Intelligence: Methods for Cooperating Ensembles in Ubiquitous Environments (AIM-CU), 30th Annual German Conference on Artificial Intelligence (KI 2007)*, Osnabrück, Germany, 2007.

[KW09] Sebastian Rollwage Kresser, Michael Klein and Peter Wolf. Collaborating context reasoners as basis for affordable AAL Systems. In *Submitted to 4rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI09)*, 2009.

[PBC06] Paolo Pellegrino, Dario Bonino, and Fulvio Corno. Domotic house gateway. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 1915–1920, New York, NY, USA, 2006. ACM.

[PG07] F. Paganelli and D. Giuli. An Ontology-Based Context Model for Home Health Monitoring and Alerting in Chronic Patient Care Networks. In *AINAW '07: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, pages 838–845, Washington, DC, USA, 2007. IEEE Computer Society.

[PVdBW+04] Davy Preuveneers, Jan Van den Bergh, Dennis Wagelaar, Andy Georges, Peter Rigole, Tim Clerckx, Yolande Berbers, Karin Coninx, Viviane Jonckers, and Koen De Bosschere. *Towards an Extensible Context Ontology for Ambient Intelligence*. 2004.

[Sch06] Andreas Schmidt. Ontology-based User Context Management: The Challenges of Dynamics and Imperfection. In Robert Meersman and Zahir Tahiri, editors, *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE. Part I.*, volume 4275 of *Lecture Notes in Computer Science*, pages 995–1011. Springer, 2006.

[WSK08] Peter Wolf, Andreas Schmidt, and Michael Klein. SOPRANO - An extensible, open AAL platform for elderly people based on semantical contracts. In *3rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI08), 18th European Conference on Artificial Intelligence (ECAI 08), Patras, Greece*, 2008.

[WZGP04] X.H. Wang, D.Q. Zhang, T. Gu, and H.K. Pung. Ontology based context modeling and reasoning using OWL. pages 18–22, March 2004.

[XWPZ04] Tao Gu Xiao, Xiao Hang Wang, Hung Keng Pung, and Da Qing Zhang. An Ontology-based Context Model in Intelligent Environments. In *In Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference*, pages 270–275, 2004.