Ontology-Centred Design of an Ambient Middleware for Assisted Living: The Case of SOPRANO*

Michael Klein[1], Andreas Schmidt[2], Rolf Lauer[3]

[1] CAS Software AG,
Wilhelm-Schickard-Str. 10-12, 76131 Karlsruhe, Germany, michael.klein@cas.de
[2] FZI Research Centre for Information Technologies,
Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany, andreas.schmidt@fzi.de
[3] Zentrum für Graphische Datenverarbeitung (ZGDV),
Rundeturmstr. 10, 64283 Darmstadt, Germany, rolf.lauer@zgdv.de

Abstract. The aim of the EU-funded project SOPRANO is to assist older Europeans to lead a more independent life in their familiar environment by means of a next generation smart home with ambient intelligence. The core of the system in each house will be the SOPRANO Ambient Middleware (SAM), which receives the user commands and sensor inputs, enriches them semantically and triggers appropriate reactions via actuators in the house. In this paper, we present a novel approach of an ontology-centred design in order to create SAM as a reliable, deterministic and economically scalable component. Thus, the starting point is the development of a context ontology focussing on the concept of a state. This OWL-Lite ontology is then used as a central reference document during the design process as well as during runtime to abstract from concrete sensor inputs and actuator outputs.

Keywords: context ontology, ambient intelligence, assisted living

1 Introduction

"Service-oriented Programmable Smart Environments for Older Europeans" (SOPRANO) is an Integrated Project in the European Commission's 6th Framework Programme. SOPRANO's aim is to enable older Europeans to lead a more independent life in their familiar environment. As an approach, SOPRANO will develop a next generation of smart homes based on ambient intelligence. Very important is the positive mindset of the project: The resulting system will not only act in problematic (e.g. fall, burglary) or emergency cases (e.g. health problems, fire) but will focus with the same attention on improving the quality of everyday life of elderly people. To achieve this goal, the SOPRANO consortium comprises 20 partners – enterprises, public bodies and research institutes from 7 European countries:

* The work published in this paper has been partly funded by the E.C. within the SOPRANO IP (www.soprano-ip.org). It does not necessarily represent the view of the E.C. or the SOPRANO consortium as a whole, and authors are solely responsible for the paper's content.

specialists in systems integration and software architecture, specialists in human factors and gerontology, local authorities for the social care of older people and experience of telecare and assistive technology. The project started in January 2007 and will last 40 months.

The technical core of the project is SAM, the SORPANO ambient middleware, which will be installed in each of the houses and provides its intelligence by receiving user commands and inputs from sensors, enriching them semantically and providing appropriate reactions via actuators in the house. Planned are sensors for e.g. smoke, temperature, door status, location of the user by Radar or RFID, its health status and so on. Planned actuators are speech synthesizers, digital TVs with avatars, device regulators (for switching devices on/off or modifying their behaviour), emergency calls to a central and more. Additionally the more static context of the house and the user shall be taken into considerations when performing concrete actions.

As SAM is supposed to be an integrating component in an open service infrastructure, a novel design approach was chosen for it to ensure semantic coherence: an *ontology-centred design*, meaning that the ontology (i.e. the formal specification of a shared data model and vocabulary) was developed as the first step in order to serve as the basis for the subsequent design decisions. In detail, the ontology is used as a blueprint for the internal data models of the components, a guideline for the communication between components by helping to define interfaces and exchanged data structures as well as a communication vehicle between the technical system and the typically non-technical user.

The paper is structured as follows: Sections 2 explains the goals and difficulties especially arising from deployment in a multitude of different homes. Section 3 introduces the general approach of SAM, its architecture and main components. After that, in Section 4 an excerpt from the SOPRANO ontology is shown and explained how this ontology is used correction. The paper concludes with a comparison with the state of the art in Section 5 and a summary and outlook in Section 6.

2 Goal and Difficulties

When setting up an intelligent system, one immediate question arises: who is providing the intelligence and by which means? As SOPRANO is not a simple tool that is only used by an IT person but a system that is fully integrated into the lives of people in order to support them in daily tasks and help them in critical situations, it is not acceptable to rely on an "unexpected intelligence" that reacts indeterministically and undeterminably. Thus, the goal of SOPRANO is to steer the reaction of the system by providing a set of rules that are executed by a more or less deterministic engine. However, to be able to economically cope with a large number of houses, these rules should be given on a semantically high level in order to be reused in different homes and situations. This means that the rules should neither be dependent on the actual outline or configuration of the house nor on the available sensors and actuators. Moreover, it should be possible that rules are entered or altered by non-technicians like the service personnel or even the assisted person (AP) herself.

Examples of such semantically high-level rules could be:

- If doorbell is ringing and AP does not recognize it, inform AP about the doorbell
- If AP watches a TV series regularly but not today, record it or remind AP
- If AP has fallen and cannot stand up again, call for help
- If AP is in a room with a too low temperature for a longer time, increase the temperature or call for help

However, there are two problems when trying to execute these rules. First, it is difficult to determine if the left side of the rule is fulfilled, i.e. if the rule fires. This is due to the fact that sensors typically only provide low-level data and are not available all the time or in every house. Examples of sensors are:

- a sensor providing the temperature of a room as integer value
- a sensor informing if a certain door in open as boolean value
- a microphone recording sound in a certain area and providing it as a stream
- a fall sensor that is currently not worn by the AP

The second difficulty applies for the actuators as devices that are able to provide rather low level actions that do not directly fit to the right side of the rules. In the best case they are integrated into a service-oriented architecture and accessible as service providers. However, not all services are available in each house, their functionality differs from house to house, or they may no be reachable. Examples for services are:

- open/close a certain window
- show an avatar on TV speaking a message
- change the volume of the TV
- send an SMS to a person

Thus, one major goal of SOPRANO will be to infer high-level context (as defined by [1]) of the AP from low-level sensor input, detect important context changes (=events), determine which rules fire, and break the initiated high-level plans down to concrete actions to be executed via service calls. This is done in the SOPRANO Ambient Middleware (SAM), to be explained in detail in the next section.

3 General Approach

The core of the SOPRANO system is SAM. SAM is an acronym for "SOPRANO Ambient Middleware" with three main components. The components are the Context Manager, the Composer and the Procedural Manager (see Figure 1). The main task of SAM is to collect context information like the change of environment, input from various sensors or user input and transfer it to the ontology and rule management system. In Figure 2, the process flow from a context change to an action is depicted. This flow is done in three steps through the three components context manager, procedural manager, and composer.

Step 1: Inferring high-level context

The *context manager* offers functionality to exploit the user's environmental information, her or his "context". Context is defined (following the definition of [2]) as the user's situation in terms of all the temporal, personal, organizational, environmental, and even global conditions surrounding her at a certain instant in time. Examples are the user's location, current time, nearby people, networked services and devices, the user's current activity and long-lasting profile, the user's environment like lighting and temperature, connectivity parameters and so on. The context manager constantly analyses incoming sensor events as well as the status of networked devices and appliances and tries to deduce higher-level context information. The result from this step are events (=changes of contextual parameters) on a high semantical level as input for the procedural manager.

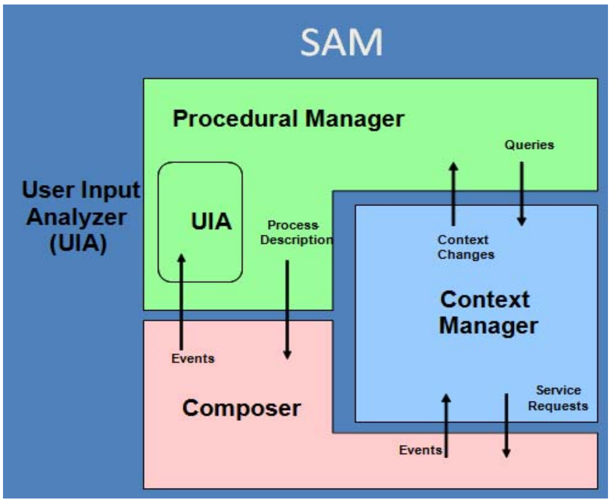

Figure 1: Cooperation of components inside SAM

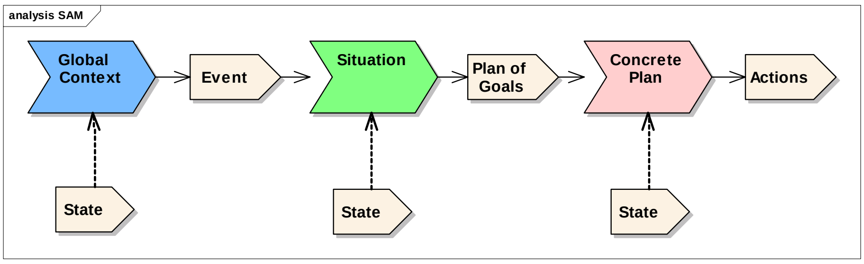

Figure 2: Flow from context change to action

Step 2: Triggering a meaningful reaction

The *procedural manager* is responsible for providing meaningful reactions to contextual changes (in the proactive case) or explicit user requests (in the reactive case which is handled by the User Input Analyzer, UIA). By analyzing the new situation, the procedural manager compiles an abstract process description based on a repository of process templates. This is a standard workflow description but contains abstract service requests instead of concrete service bindings. It can be parameterized with contextual variables, is based on pre-defined templates and is annotated with context-aware metadata. The procedural manager can obtain additional state info by invoking ad-hoc queries to the context manager. The result from this second step is a "plan of goals" which serves as input for the composer.

Step 3: Executing the plan via service invocations

The *composer* has two objectives. First, it serves as SAM's interface to the "real world" with sensor information, abstract controlling of actors and in- and output devices. All incoming and outgoing service calls are handled by it. The ontological concepts are used to describe the environment in the context manager and are used to formally specify the scenarios. Second, it receives the "plan of goals" from the procedural manager, intelligently searches, compares, composes and parameterizes suiting concrete services in order to be able to execute the process in a concrete context-aware manner and execute these via actuators.

4 The SOPRANO context ontology and its use

With SOPRANO's goal being the establishment of an open infrastructure for a variety of sensors, actuators and services in an intelligent home environment, we need an infrastructure with a sufficient degree of semantic coherence in order to enable smooth interoperability. Ontology-centred architectures as describe e.g. in [3], i.e., architecture in which ontologies are of primary importance for the specification of contracts among components, allows for (a) a descriptive representation of the common system vocabulary and (b) supporting varying degrees of abstraction (among potentially other ways of exploiting background knowledge). From the previous section, we have seen that the focal point of the SOPRANO system is the context of the assisted person: sensors are used to acquire information about it, the context manager uplifts it and the system reaction is triggered by context changes.

Therefore, the SOPRANO ontology is built around the notion of context. There have been several approaches to ontology-based modelling of context in an assisted living scenario like [4] or COMANTO [5] (a good survey is given in [6]), so which are the specific requirements of SOPRANO? From an analysis of usage scenarios, we found out (similar to [7]) that

- we need to represent not only the current context, but also past contexts (e.g. the system will respond differently if the assisted person has fallen recently);
- we need to be able to represent uncertainty as the inference process of high-level context information is not unambiguous, but rather based on possibly contradicting heuristics;
- most problems in dealing with context information cannot be adequately solved with the reasoning paradigm and that we should rather use the query paradigm (i.e., we don't need to classify, but rather have complex computations like computing the rate of change etc.)
- the main benefit of ontological modelling is the possibility to deal with different levels of abstraction (inheritance)

From these requirements, we chose a lightweight approach to ontology modelling, mainly taking OWL-Lite as a modelling formalism. As a design principle, SOPRANO has taken primarily a state perspective on modelling the context of an assisted person. State has turned out to be a shared concept among the different involved components:

- sensors yield state information or state update messages,
- context-aware system response is triggered by state changes (= events),
- specified by a desired state (= goal)
- and constrained by state changes.

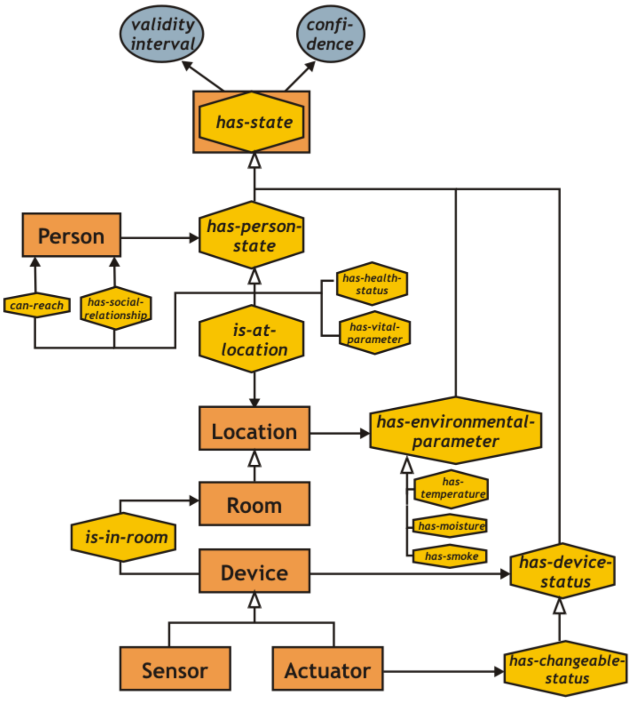

Figure 3: Core concepts and properties of the SOPRANO context ontology
(in an extended ER notation from [8])

Instead of describing states as an instance set with a structuring concept hierarchy, this state-driven perspective is represented as (object) properties and a respective hierarchy (following the context modelling approach in [7]). Instances of these properties are represented with temporal metadata (transaction time and validity intervals) as well as confidence. The temporal perspective is needed as we cannot solely rely on the current state, but have to take into account past state information as well. The top-level structure of the ontology is given in Figure 3. The main sub-properties of the central *has-state* property are:

- *has-person-state* for all state information that directly depends on the person, including the assisted person, but also carers, neighbours, among others (e.g., vital parameters, health status, social relationships, location);

- *has-environmental-parameter* for all location-dependent state information (e.g., outside conditions like weather, moisture etc. and room-dependent conditions like temperature);
- *has-device-status* for all device dependent state information (like power on/off, current set-points), including sensors and actuators. Actuator states are differentiated by marking them as changeable (*has-changeable-state*)

How is this ontology used by the different components of the SOPRANO system? The contract with sensor components specifies that they provide basically state update messages (e.g. temperature is 23 degrees, fridge door is open) to the context manager. This can be done via push mechanisms (the sensors sends state information to the context manager) or pull mechanisms (the context manager can query the sensor). The context manager updates the global context with this information and possibly aggregates it to other state information.

The Procedural Manager registers with the Context Manager for changes to the state (e.g. AP has fallen, temperature in kitchen has fallen from medium to low, AP has not left apartment for more than 24 hours). Such continuous queries can contain additional semantic classification rules (like classifying absolute temperature into subject temperature interpretations as low/cold or high/hot). Based on these context changes, the procedural manager can select predefined goal templates whose basic building blocks are made up of goals as desirable target states (e.g. *neighbour informed*, if not possible or no reaction after 15 minutes: *relatives informed*, if not possible *service centre informed*). Based on these abstract target states, the composer selects appropriate services described according to the state changes they can achieve.

This state-centric core of the SOPRANO ontology can be expressed in a subset of OWL Lite and thus allows for efficient implementation on top of relational databases. Subcomponents, though, might benefit from representing additional background knowledge in the ontology, e.g., for context aggregation. Determining whether an assisted person is in a risky situation (which should trigger proactive system behaviour, e.g. speech output to the AP) can require additional information.

5 Related Work

SOPRANO as an integrating project builds upon a body of research on the subject of smart homes and ambient assistive technologies. Some recent examples in the domain are the DAIDALOS project [5] or the AMIGO project [9]. The main difference of the SOPRANO approach is that we want to bring together a service oriented approach like [11] with ontologies on an architectural level. This means that the ontology is used as a contract between all components of the system in order to ensure semantic coherence. Most comparable ontology-based systems primarily use the ontology in the context of reasoning while in SOPRANO we use the ontology to for the communication with sensor and actuator services, for expressing the state of the AP and the environment, for events, for system responses and for describing service capabilities. To make this work, we cannot rely on highly expressive ontology formalisms (because not all problems can be expressed well in terms of reasoning), but rather stick to formalisms with a low level of expressiveness.

6 Summary and Outlook

In this paper, we have presented the initial architectural work within the EU project SOPRANO which aims at helping older Europeans to lead a more independent life. The developed smart home is based on a core called SAM, the SOPRANO ambient middleware. We have shown a novel ontology-centred design approach for this component in order to get an intelligently acting, but also deterministic and economically scalable module. A shared context ontology focussing on state properties as central concept is the reference for all other parts of the system, both used both during design time to define the interfaces and internal data models of the various components as well as during runtime for inferring high-level context from low-level sensor input and describe actuator output in a general, device-independent manner. First prototypes of SORPANO houses basing on SAM are expected at the beginning of 2008. The evaluation of the developed concepts is also planned in this year, both in a large-scale field trial in 300 homes with restricted functionality and a fully-functional trial in dedicated smart home environments.

References

1. Winograd, T. Architectures for context. Human-Computer Interaction 16 (2001)
2. Dey, A. K.: Understanding and Using Context. Personal Ubiquitous Comp. 5(1), 2001, 4-7
3. Schmidt, A.: Impact of Context-Awareness on the Architecture of E-Learning Solutions. In: Pahl, C. (ed.) Architecture Solutions for E-Learning Systems, IGI Publishing (2007)
4. Preuveneers, D., Van den Bergh, J., Wagelaar, D., Georges, A., Rigole. P., Clerckx, T., Berbers, Y., Coninx, K., Jonckers, V., De Bosschere, K. (2004) Towards an Extensible Context Ontology for Ambient Intelligence. Proc. of EUSAI 2004, Springer, LNCS vol. 3294, 148-159
5. Roussaki, I., Strimpakou, M., Pils, C., Kalatzis, N., Anagnostou, M. Hybrid context modelling: A location-based scheme using ontologies. Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06) (2006)
6. Baumgartner, N.; Retschitzegger, W.: A Survey of Upper Ontologies for Situation Awareness. In: Proceedings of Knowledge Sharing and Collaborative Engineering (2006)
7. Schmidt, A.: Ontology-based User Context Management: The Challenges of Dynamics and Imperfection. In: On the Move to Meaningful Internet Systems 2006, LNCS vol. 4275, Springer (2006), 995-1011
8. Schmidt, A., Kunzmann, C.: Towards a Human Resource Development Ontology for Combining Competence Management and Technology-Enhanced Workplace Learning In: On The Move to Meaningful Internet Systems 2006, Part III, LNCS vol. 4278, Springer (2006), 1078-1087
9. Magerkurth, C., Etter, R., Janse, M., Kela, J., Kocsis, O., Ramparany. F.: An Intelligent User Service Architecture for Networked Home Environments. Proc. of the 2nd Int, Conf. on Intelligent Environments, Athens, Greece, July 5-6 (2006),. 361-370
10. Wu, C., Liao, C., Fu, L.: Service-Oriented Smart-Home Architecture Based on OSGi and Mobile-Agent Technology, IEEE Trans. on Systems, Man and Cybernetics 37(2) 2007