Modeling the Home Environment using Ontology with Applications in Software Configuration Management

Elena Meshkova, Janne Riijijärvi and Petri Mähönen
Department of Wireless Networks, RWTH Aachen University
Kackertstrasse 9, D-52072 Aachen, Germany
Email: {eme, jar, pma}@mobnets.rwth-aachen.de

Christoforos Kavadias
Teletel
Athens, Greece
Email: C.Kavadias@teletel.gr

Abstract—The rapid development of the network technologies has enabled the construction of heterogeneous home networks. However, the control of the smart home environment that ensures the conflict-free, up-to-date and smooth functioning of the home environment is not trivial. It is necessary to track the state of the numerous devices which compose the home environment, as well as the respective services installed on them, to ensure conflict-free seamless operation of the system. In this paper we present an ontology-enabled knowledge base for home networks. The knowledge base allows to depict complex relations between devices and services working in a home environment, for example issues to be considered before installing a new software version on a certain device, such as the user profile, interference with existing devices, etc. The knowledge base is a part of the COMANCHE system, which aims to provide a software configuration management infrastructure which can be used in the home automation domain. To the best of our knowledge, the presented system provides a first comprehensive framework for reasoning about services in the home environment realized in close collaboration with the relevant industry.

I. INTRODUCTION

The research and development in the area of smart homes is booming. The advantages in networking and computer miniaturization have allowed the construction of complex heterogeneous networks which are an inevitable part of a ubiquitous environment. However, the real-life deployment of intelligent home faces many challenges. W. Edwards and R. Grinter [1] have pointed among others the following issues: the partial, non-instantaneous, introduction of new technology at home, interoperability issues, high reliability demands and need for self-configuration.

The solution of the above issues is impossible without a solid model of context description of the home environment. One can distinguish between several basic context models [2]. Application-oriented approach, for example exercised by HP's CoolTown [3], concentrates on modeling and representing content for only specific application. Often these models lack formality and expressiveness. Model-oriented approach typically uses conceptual modeling to represent context, the examples are Entity-Relationship model and UML diagrams [4], [5], which might be suboptimal for constant run-time update. Ontology-oriented approach allows to construct a knowledge base containing information on the certain environment which context model is represented as an ontology. The use of ontology allows creation of very flexible models on which deductions and reasoning can be done. Ontologies are also the core component of the Semantic Web [6] concept.

We have chosen the ontology-oriented approach to perform context modeling of the home environment as it suits the best for the scalability and web-orientation requirements that a large distributed software management system such as COMANCHE [7] faces. COMANCHE stands for Software CONFIGuration MAinagement framework for Networked services environments and architectures incorporating ambient intelligence features. Its main purpose is to create a service-oriented system that allows to simultaneously install, configure and monitor services in multiple heterogeneous home environments ensuring integrity and smooth functioning of home networks according to the user needs. In this paper we present an ontology-enabled knowledge base (KB) that models a home environment in terms of services it provides and the respective hardware, software and network components. It also communicates data concerning users of home environments and providers of software components and hardware devices to cope with the security issues. The suggested ontology model is, to our knowledge, the first that distinguished between abstract and real service implementation in the context of the home environment, as well as allows to keep track and react on numerous events in the home network.

The rest of the paper is structured as follows: in Section II we discuss technologies that forms the foundation of our work. The general description of COMANCHE architecture is given in Section III. The home environment ontology that is used in the knowledge base is described in Section IV along with the tools that are used for its development. Section V concludes the paper.

II. ONTOLOGY TECHNOLOGY OVERVIEW

Computer scientists use the term ontology to specify a conceptualization [8] in the context of knowledge sharing. By conceptualization we understand an abstract, simplified view of the environment we want to represent. Every knowledge
The primary goal of the COMANCHE framework is management of the home environment services. One of the pilot scenarios of the system usage is installation of the new devices, for example a washing machine, and later expansion of the services it provides. In our scenario the user, Alice, installs a new washing machine at home. The machine connects to the home gateway and notifies it that a new device has appeared in the network. The gateway does not have the COMANCHE device agent for this washing machine model and requests the COMANCHE server for the name of the appropriate software. This information is fetched from the central knowledge base (KB). Then the gateway directly inquires the provider for the needed software, downloads and installs it. The next step is to update the part of the KB that stores information about this user’s home environment with the information concerning the new washing machine, for example its hardware capabilities and default software installed.

After a while Alice wants to install new services onto the washing machine, for example a power management function and a new advanced washing program. First, in order to enable the installation the KB has to be inquired to estimate interdependencies for the desired services. For example, it needs to be ensured that the power management service is compatible with all other devices at the home network, so a common power management can be enabled. For the new washing program the KB has to check if the washing machine has the hardware capabilities to run the software and has installed all the services the new washing program requires, like the basic washing services. Additionally, the framework has to ensure the security of the user home environment, for example allow installation of components only from the trusted providers.

B. General architecture

The simplified architecture of the COMANCHE framework is shown in Figure 1. The framework consists of local modules executed at individual home networks and software run centrally at one or several servers on the Internet side. The local modules include a Software Configuration Management (SCM) Gateway and Device Agent functional entities. The SCM Gateway is the software component running on a appropriate processing- and communications-capable device located in the Home Environment. Most probably this will
be a dedicated device like a home gateway that is capable of handling all of the internal home network traffic and regulating network interactions with the external world. The purpose of the SCM gateway is local, home network scale, coordination of software configuration and providing notification of state changes in the home environment, for example of the user requests or malfunctioning of the devices.

Each device that is part of the COMANCHE Home Environment has to have an SCM Device Agent associated with it. A Device Agent is a software module running on either the device itself or on another processing- and communications-capable device located in the Home Environment, for example a home gateway. The purpose of the Device Agent to interact with both the SCM Gateway and the relevant device and to forward information between them in the format acceptable to both. The Device Agent translates the COMANCHE signaling into the internal, often proprietary, protocol format that is understandable by the home device, and puts the device’s messaging in the format appropriate for the SCM gateway.

The server hosts the following units: an ontology-enabled SCM Knowledge Base that stores knowledge about devices, services, users and produces of home environments, a SCM Engine that interacts with the KB and the Service Consistency Validator that can perform in-depth analysis of the services structure. There also exists the Identity Provider which main aim is to establish trust in the context of the COMANCHE architecture between all entities involved like users, providers and devices, by authenticating everybody and performing identity federation in consistence with the privacy preferences of users.

Additionally, the framework interacts with two external independent entities. The Software Service Providers deliver services to the home users and allow to download and deploy software components onto home devices. The Hardware Providers give description of devices deployed at home. Further we describe the most relevant to our work components of the COMANCHE architecture in more detail.

C. Software Configuration Knowledge Manager

Software Configuration Knowledge Manager is the central piece of the COMANCHE system. Its main components are SCM Engine and SCM Knowledge Base. The SCM Engine reacts on the requests raised by system entities. For example it is the job of the SCM Engine to formulate a request to the knowledge base and get the list of services that need to be deployed on the washing machine prior to the installation of the new washing program. It is also the job of the SCM Engine to evaluate the obtained answer and either to forward it to the respective home environment (SCM Gateway) or make an additional request to the KB or the Service Consistency Validator. The intelligence of the SCM Engine largely depends on the structure of the knowledge base, i.e. its ontology.

The SCM Knowledge Base receives and processes requests from the SCM Engine, delivering conflict-free knowledge instantiations. In order to achieve this some additional, expert-system level, mechanisms of the KB are employed that are capable of resolving potential conflicts identified in the knowledge instantiations obtained according to the SCM Engine request. In other words the SCM Knowledge base consists of the COMANCHE ontology as well as its conflict-free instances. It provides/updates data according to the SCM Engine request. Additionally, SCM Knowledge Base interacts with the Identity, Software Service and Hardware Providers through the SCM Engine in order to allows the providers to publish their data in the KB. For example Service Software Providers can announce services they suggest, the interrelations between them and the respective software components that realize these services.

IV. Home Environment and Context Ontology

There exist several ontologies that describe smart home environments [2], [12], [13], [14]. However, these ontologies do not allow to model complex relations between services, the software that realizes them, users, hardware and the software/hardware providers. Our ontology models these complex relations, and it even incorporates some elements of the actual home network modeling like the bandwidth requirement of different software services. Additionally, the suggested ontology is developed according to the requirements received from the leading home appliances producers like Indesit and Gorenje. It is interesting to note that in spite of the complexity of the described environment our home ontology is still manageable due to the heavy hierarchical structuring of both classes and properties.

The ontology is the skeleton that enables storage of information in the format which allows execution of the main task of the developed framework: automatic service selection, composition, deployment and interoperation. In order to fulfill this task the knowledge of the home environment, the user and business domains are necessary as they provide the constraints and optimization criteria. Having this information the ontology-enabled knowledge base can provide prerequisites and consequences of application of individual software services in a certain home network.

The COMANCHE ontology is composed of the Service Ontology, the Home Environment and Context Ontology and the Business Domain and User Ontology. The Service Ontology relies on the OWL-S specification [15], its purpose is to describe the complex interrelations between services, using besides basic logic operators also the more advanced ones as if and loop statements. The Service Ontology, as well as the COMANCHE project as the whole operates with two types of services. The Software Service represents a collection of technical functions installed on a device and implemented through the use of software. A device can host multiple software services. Examples of the software services are a firmware that can be installed on a home appliance or a software module like a specific washing machine program. The Internet Service represents a collection of functions to be accessed through the Internet and can be called remotely. Example of the Internet Services is an execution of an arbitrary Web-Service like a remote diagnostics and maintenance procedure. The Business Domain and User Ontology is used to represent and exploit
information relating to business relationships, user identity management, and user preferences particularly in terms of privacy. The Home Environment and Context Ontology is described in the next subsection.

A. Structure of the ontology

The structure of the suggested Home Environment and Context Ontology is shown in Figure 2. The Home Environment and Context Ontology is used to describe the home environment of each user. The ontology mainly includes the description of the devices installed at home, the service and the software that realized these services. Additional classes like Event class or User class help to describe the home environment more fully, but are complementary to the main purpose of this ontology: service modeling in a home environment. For example the Event class specifies events that a device or a user might encounter or produce. The ontology also incorporates multiple subclasses that are not shown in the basic ontology diagram. Subclasses are created to reflect rules imposed onto the ontology. For example we create a subclass of the Device Function class that holds all the functions that a washing machine of a certain type might perform. This enables the inference engine to automatically determine which functions can be installed on the washing machines of different types. Similarly, the creation of subclasses of the Software Service class that contain services already installed on the individual devices allows to automatically deducing services that have additionally to be installed on these devices. Further we shortly describe each of the main classes of the Home Environment and Context Ontology, as well as relations between them.

B. Home Environment class

The Home Environment class contains basic information about a home environment. It provides data concerning the users of the home environment and all the devices installed at home. Additionally, separate classes are created to represent the Home Gateway device and the SCM Gateway service, due to the special nature of these entities. All events that happen in the users home are known to the home environment through the SCM Gateway. These events can be handled either by the SCM Gateway or the SCM Provider. For each home a separate instance of the Home Environment class is created.

C. Device domain classes

There exist four classes that are used to describe a device: Device, Device Type, Device Function and Device Function Installation classes. The instance of the Device class represents a specific appliance placed in the home environment. For example it can be the user’s washing machine. The profile of the device includes its model/type, the link to the event depositary, the list of functionalities and the respective software services that it can provide or wants to install, and the link to the specific SCM gateway that handles the appliance.

Home Gateway is one of the child classes of the Device class. A home gateway handles all the incoming and outgoing network traffic, therefore it has to support extended security functionality, handle heavy traffic and, more importantly, have a special secure registration procedure with the SCM Gateway compared to the ordinary home device. The registration procedure of the home gateway to the SCM gateway is to be designed so that, from one side, it requires minimal user interference and, from another side, it does not compromise the homes security.

The Device Type class describes the model of the device. The class contains the list of functionalities a device can support, the producer’s name, the description of the hardware and software platforms that run on the device, as well as the list of supported network interfaces. Each instance of the Device Type class is classified according to the Device Category to which it belongs. The Device Category class is a simple enumeration of instances such as Washing machines category, Ovens category, Sensors category and Home gateways category. Depending on the value of the Device Category the Device Type class is further subdivided into the classes that distinguish between various types of devices, such as washing machines, ovens, dishwasher, sensors, etc.

The Device Function class contains the abstract description of the services that devices are able to provide. The Device Type class lists all the functionalities that a specific device model can support. The Device class contains the list of services that are already installed or need to be installed. In order for a device to perform any service, the software implementing this service is to be installed. The mappings of the abstract functions to the real software are done using the Device Function Installation class.
D. Platform and Network classes

In order to be able to determine if a service can be installed on the certain type of device, the hardware and the software profiles of the model have to be stored. The ontology classes Device Software Platform and Device Hardware Platform provide this capability. The Device Software Platform class stores information concerning the operating systems, virtual machines and other supporting software installed on the device. The Device Hardware Platform class provides information on the hardware profile of the device. It stores such parameters as RAM, ROM, processing capabilities, peripherals, UI capabilities, etc. The Network Interface class is used to specify the network capabilities of the specific device type. The Network Interface class stores information about the network interface used by the device, the available bandwidth, the driver version installed, etc.

E. Provider classes

The Provider class contains information about providers and producers of the devices and the software services. The two child classes of the Provider class are Software Service Provider and Device Provider classes. The Device Provider class contains the list of the device models that the specific producer offers. The Software Service Provider class provides the list of offered software services. Both classes store additional information concerning the trustworthiness of the particular provider.

F. Event class

The Event class is created to gather information, keep statistics and react on the events occurring in the home environment. The outcome of any action initiated by the SCM is registered by the instance of the event class. For example if a new service is successfully downloaded on to a device, the event of class Success is raised. If the operation fails the event of class Failure is signaled, for example Channel Busy. The Event class is divided into two classes Fault Event and Success Event. These classes signal either a success or a failure of a certain operation. The instances of the classes differ in the status codes defined in the HTTP manner. The example Fault Event instances include Service Inaccessible, Connection Breakdown, Device Incompatible, Device Inaccessible, Channel Busy and Low Bandwidth events. The Success Events instances can be Connection Established and Operation Completed events. Additionally, all of the Event instances store the time of their creation for the statistics gathering purposes. The events are stored at the SCM gateway and can be accessed there for further processing.

G. Software service classes

The Software Service class and its child the SCM Gateway class are one of the most important classes in the whole Comanche ontology. This class actually leads to the whole Service Ontology which describes the relations and interdependencies between different services in the OWL-S manner. In our prototype implementation we have only realized the most basic of the relations like the requirement and exclusion lists, i.e. the list of services that need to be installed onto the device before deployment of the specific service as well as the list of incompatible services. Additionally, the Software Service class includes the requirement of the software module to the hardware platform, a list of devices where this program is installed and a link to the provider.

The SCM Gateway is a child of the Software Service class. The instance of this class depicts the SCM Gateway that controls the devices operating in the home environment. Typically, the SCM Gateway is installed on the home gateway device, however it is possible to put it on any other device that has sufficient capabilities and operates constantly in the home. Additionally, SCM Gateway stores and handles the events occurring in the home network.

H. Summary

The Home Environment and Context Ontology results in rather complex structure that includes, besides the upper level classes described above, also numerous automatically generated subclasses and multiple rules that dictate distribution of objects between classes. As the result the suggested ontology is able to install and manage both abstract services and the respective software on the home devices ensuring secure and conflict free functioning of the home environment. The simple example of the inference performed on the KB is given in Figure 3. The KB can also respond to changes in the home environment by updating old or installing new services, resulting in the seamless configuration of the home network. Additionally, the system allows to keep track of the events.
happening in the home network in order, for example, to assist the software/hardware providers in the remote maintenance of the home.

I. Tools

Currently we use several tools in our work, most of them are realized on Java or have Java-compatible interfaces. For the realization of the knowledge base we are using Protege [16], FACT++ [17] and Jena library [18].

Protege is a visual environment for creating and visualizing ontologies developed at Cambridge University. It features a reasoning API that is used to access an external DIG description logic interface [19] compliant reasoner, like FACT++, thus allowing to infer about classes and individuals in the ontology-enabled knowledge base. The availability of the DIG interface also enables the use of automatic reasoners that can provide various inferring services, like estimating the consistency of classes and estimating objects that belong to a particular class. We have chosen FACT++ as the automatic reasoner to be used in our project because this is an open source software and judging from our experience it provides the best performance among most well known open reasoners.

The standard reasoners like FACT++ or Pellet [20] are very comfortable to use for test development. However, they have their disadvantages. First of all they still have a limited functionality, i.e., some rules can not be evaluated with the existing open-source reasoners. Another problem is that they are not as fast as needed. Therefore, extra functionality has to be realized using either Jena library or existing Protege plug-ins like SWRL [21] which relies on proprietary programming language Jess [22] designed for development of expert systems. In our work we have decided for Jena as it is an open source library, through it requires a lower-level programming compared to Jess.

V. Conclusions

We have presented and realized a scalable ontology-enabled knowledge base for home networks that provides the means for effectively conceptualising, organizing, and exploiting the tremendous amounts of attribute information, pertaining to the management of home services. The ontology incorporates the requirements for white goods appliances provided by our industry partners Indesit and Gorenje. The current version of the KB depicts services and devices installed at home, allows to monitor interconnections between services and ensures integrity of the home network. It can be also easily extended to incorporate new rules and new sorts of data. For example it can be further expanded to control the traffic in a home network to ensure maximal throughput and QoS of home devices. The KB is a part of the COMANCHE system that utilizes a component-based software architecture in order to effectively address the engineering and run-time management of reconfigurable software for ambient intelligent networked services environments.

At present we are starting to integrate the ontology-enabled knowledge base with the rest of the COMANCHE framework and test the system with real white goods appliances like washing machines from Indesit and ovens from Gorenje. As soon as the integration work is finished we are planning to make the Home Environment and Context Ontology with sample instances publicly available.

ACKNOWLEDGMENT

We would like to thank DFG (Deutsche Forschungsgemeinschaft), European Union (COMANCHE-project) and RWTH Aachen University for financial support. Additionally we would like to thank all the partners of the COMANCHE consortium for fruitful discussions. PM and JR are also acknowledging the partial financial support from UMIC-excellence cluster at the RWTH Aachen University.

REFERENCES