First Demonstrator of HYDRA Middleware Architecture for Building Automation

Martin Sarnovský[1,2], Peter Kostelník[2], Peter Butka[1,2],
Ján Hreňo[2], Dáša Lacková[2]

[1]Faculty of Electrical Engineering and Informatics, Technical University of Košice,
Letná 9, 040 01 Košice, Slovak Republic
{martin.sarnovsky, peter.butka}@tuke.sk
[2]Faculty of Economics, Technical University of Košice, B. Nemcovej 32,
040 01 Košice, Slovak Republic
{peter.kostelnik, jan.hreno, dasa.lackova}@tuke.sk

Abstract. This paper describes the IST-2005-034891 project HYDRA ("Networked Embedded System Middleware for Heterogeneous Physical Devices in a Distributed Architecture") funded within the FP6 IST Programme. HYDRA project aims at development of middleware for intelligent networked embedded system based on service-oriented architecture, deployable on both new and existing networks of distributed wireless and wired devices. The embedded service-oriented architecture will provide interoperable access to data, information and knowledge across heterogeneous platforms. These devices and their local networks will also be interconnected through broadband and/or wireless networks. An implemented HYDRA middleware and a toolkit will be validated in real end-user scenarios in three different user domains: Facility management (smart/intelligent homes), Healthcare, and Agriculture. The following description of the architecture is based on a functional viewpoint of the middleware in which we describe the identified layers and components and how they relate to each other. The different components are described in detail and some more details of the sub-components are added. In addition to this, we have identified the requirements which have implicit or explicit relationship to the component. In the following chapter we present a first scenario in domain of building automation which we have researched in detail to explain what could be done with the architecture.

Keywords: ambient intelligence, networked embedded systems, middleware, semantic technology, service-oriented architecture, ontology modelling

1. Introduction

The HYDRA project is addressing the problem, which is frequently faced by producers of devices and components - the need for (which is actually becoming a trend) networking the products available on the market in order to provide higher value-added solutions for their customers [3]. This requirement is implied by citizen centred demands requiring intelligent solutions, where the complexity is hidden behind user-friendly interfaces to promote inclusion. The vision of the HYDRA project is ambitious:

"To create the most widely deployed middleware for intelligent networked embedded systems that will allow producers to develop cost-effective and innovative embedded applications for new and already existing devices."

To put it in practical terms: In the ambient world of the near future, interconnected intelligent devices will surround us, at home, work, or while travelling. These devices and their local networks will also be connected to the outside world through broadband and/or wireless networks [2]. Numerous services to support us in our personal life will be provided through these ambient devices and over the connection to the outside world. To adapt to our personal lifestyle, and to offer the right service at the right time in the right place, such services will rely on the use of private data - which means putting emphasis also on security and privacy. It is expected that the HYDRA will contribute to this scenario.

HYDRA will develop a middleware based on a Service-oriented Architecture (SOA), to which the underlying communication layer is transparent. Hydra middleware will be designed to run on a variety of stationary and mobile devices. The middleware will include support for distributed as well as centralised architectures, security and trust, reflective properties and model-driven development of applications. It will be deployable on both new and existing networks of distributed wireless and wired devices, which operate with limited resources in terms of computing power, energy and memory usage. It will allow for secure, trustworthy, and fault tolerant applications through the use of novel distributed security and social trust components and advanced Grid technologies.

The embedded and mobile Service-oriented Architecture will provide interoperable access to data, information and knowledge across heterogeneous platforms, including web services, and support true ambient intelligence for ubiquitous networked devices. Furthermore HYDRA will develop a Software Development Kit (SDK), which will be used by developers to develop innovative Model-Driven applications using the Hydra middleware. Middleware and connected devices should enable developers to implement applications that depend on and adapt to context information. In particular, the developers stressed the acquisition and management of spatial context information that allows for locating devices attached to the system and for the positioning of people and assets. The HYDRA project will validate the middleware, the SDK toolkit in real end-user scenarios in three user domains.

2. Related Work

SOCRADES project is to create new methodologies, technologies and tools for the modeling, design, implementation and operation of networked hardware and software systems embedded in smart physical objects. SOCRADES project presented the idea of integration of SOA-ready embedded devices into the enterprise systems [5]. Authors are assuming that networked embedded devices can be SOA-ready and should offer their functionality via a web service. *UbiSec&Sens*[1] will provide a comprehen-

[1] www.ist-ubisecsens.org

sive architecture for medium and large scale wireless sensor networks with the full level of security that will make them trusted and secure for all applications. The main goal of the project can is to provide a complete toolbox of security aware components for wireless sensor network application development. The objective of S3MS[2] project is to create a framework and a technological solution for trusted deployment and execution of communicating mobile applications in heterogeneous environments. S3MS would enable the opening of the software market of nomadic devices (from smart phones to PDA) to trusted third party applications beyond the sandbox model, without the burden of roaming trust infrastructure but without compromising security and privacy requirements. AMIGO[3] project's main objective is to research and develop open, standardized, interoperable middleware and intelligent user services for the networked home environment, which offer users intuitive, personalized and unobtrusive interaction by providing seamless interoperability of services and applications.

3. Scenarios for usage of Hydra

The HYDRA middleware addresses two different types of users:

- Developer users, who will use the Hydra middleware to develop their applications,
- End-users, who will use Hydra applications developed by the developer users.

Both types of users are involved and studied in the project. One of the first tasks in HYDRA project was creating scenarios of end-user behaviour and interaction with platform functionality in three different user domains, which were selected for pilot applications within the project – Building automation, Healthcare, and Agriculture. Scenarios were created by using the IDON method that consists of two parts [1]:

- Scenario development using experts and based on knowledge and systematic analysis. The aim is to develop four mind-challenging scenarios for each user domain by mixing inevitable trends with creative fiction.
- Scenario deployment – in this part technical experts and project decision makers interpret the scenarios and extract a framework for the functional and trust and security requirement specifications.

The scenarios developed provide coherent, comprehensive, internally consistent descriptions of plausible futures built on the imagined interaction of key trends.

[2] www.s3ms.org
[3] www.amigo-project.org

4. Scenario description

For the first pilot application the Building Automation domain was selected, as the only domain to investigate [1]. The scenario (developed by In-Jet project partner) shows a fictitious future situation, to:

- convey a better understanding of the system;
- discover potential building blocks of the system;
- analysis of the interaction of the end-user with the system.

The resident is living in a new flat in the "Krøyers Plads" housing complex in Copenhagen. In addition to the usual set of automatic lamps, computer and wireless network, the flat is also equipped with an automatic heating system. While the resident is at his office, he receives an alert from his "*Hydra Building Automation System*" (HBAS) that the heating system has broken down. Since the temperature has reached sub zero level, HBAS categorized it as an emergency situation and tried to contact the resident until he replied to the alert.

Since the resident has a contract with the service provider of the heating system to send out a service agent to repair the system in case of a break down, the resident sends a repair order to the service provider. The service provider sends out a service agent to the flat. The service provider has transferred the appropriate credentials to enter the house and to repair the heating system, to his the service agent´s PDA. When the service agent arrives at the complex, he authenticates himself to the door and is given access to the resident's flat after successful validation. The service agent checks the logs in the heating system to identify the errors and uses the online help of the service provider to fix the problem.

After finishing his work, the service agent leaves the flat and the HBAS system informs the resident that the heating system is working again and the service agent has left the flat.

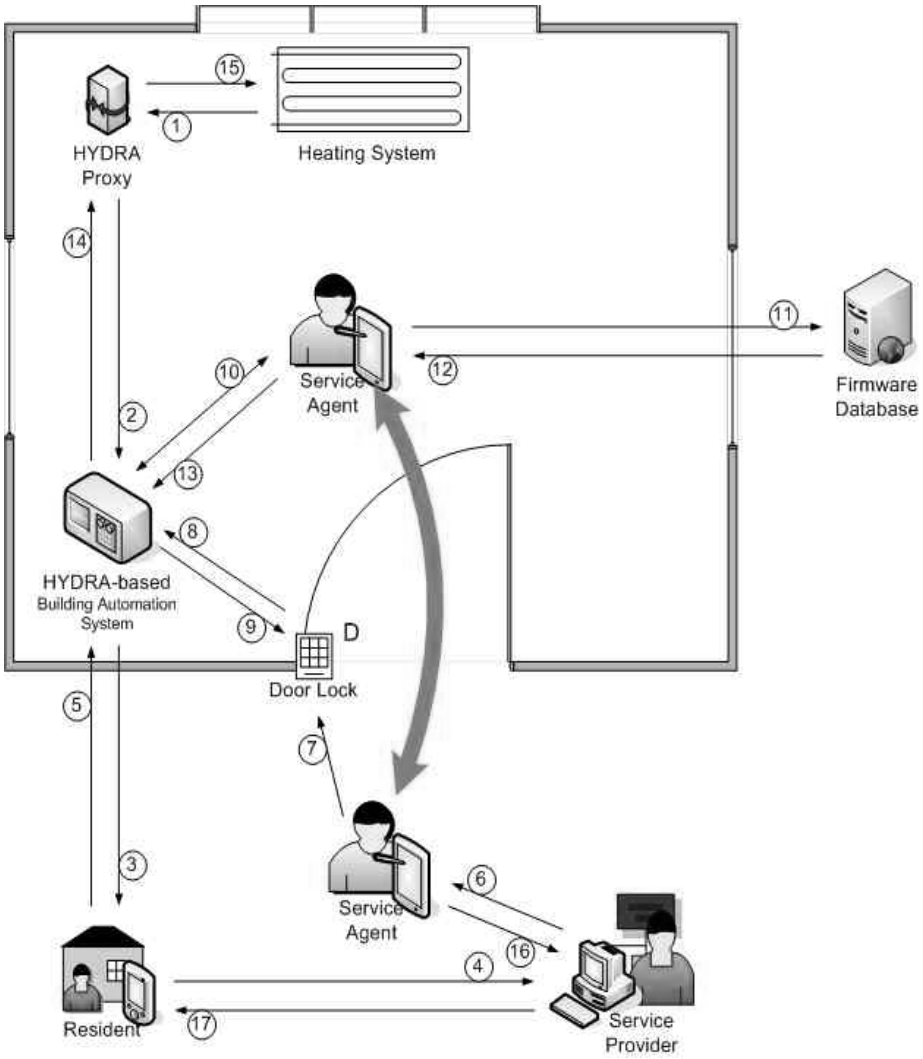

Fig. 1 Diagram illustrating the main steps of the scenario.

Legend to Fig.1:

1. Send error to HYDRA proxy
2. Send error message to HYDRA-based HBAS
3. Send error message to resident
4. Send service request to Service Provider (including credentials)
5. Send error confirmation with credentials for Service Provider
6. Send service order with encrypted credentials to Service Agent
7. Authentication at door using credentials
8. Send credentials to HBAS

9. Send message to open door lock
10. Check error
11. Send request for firmware update to a firmware database
12. Download firmware update
13. Upload new firmware to HBAS
14. Upload new firmware to HYDRA Proxy
15. Install new firmware
16. Service Agent sends error report to Service Provider
17. Service Provider sends the bill to resident

The runtime platform model defines the requirements on the computing nodes and which software components run on each node. There is six identified Hydra enabled nodes in our demonstrator scenario and two non-Hydra devices. The main communication method is wireless and is based on Wireless LAN. The other wireless protocols like GPRS or UMTS could be used to communicate with users outside the proximity of the HBAS but this would be seamlessly handled by the Network Manager and the Context Manager but will not be supported in the first demonstrator. To show the handling of several transport protocols we use a Bluetooth communication with the door proxy and the wireless communication with the HBAS will only be supported if the service agent has successfully authenticated himself.

5. Software architecture

The Functional Structure model is divided into two parts: Application Elements and Device Elements. Both elements differ in the following aspects:

- Power of the machine
- Intended purpose of the components
- Target developer user

Application elements describe components that are usually deployed on hardware which is performance-wise capable of running the application that the solution-provider creates. This means these components are meant to be run on powerful machines. They have been put together and configured to work together with other software in order to support a specific application such as building automation by a specific developer e.g. system integrator.

Device elements describe components that are usually deployed inside HYDRA-enabled devices so we take into account that they could be deployed in small devices which have limited resources in terms of e.g. processing power or battery life. Those components have a limited set of functionality but could also be deployed on another machine acting as a proxy for e.g. a mote where it would be highly unlikely that those managers would ever be deployed on such a resource-limited device. They have been

put on the device by a device manufacturer to provide certain functions irrespective of which application is using the device.

Figure 2 gives a structural overview of the middleware layers:

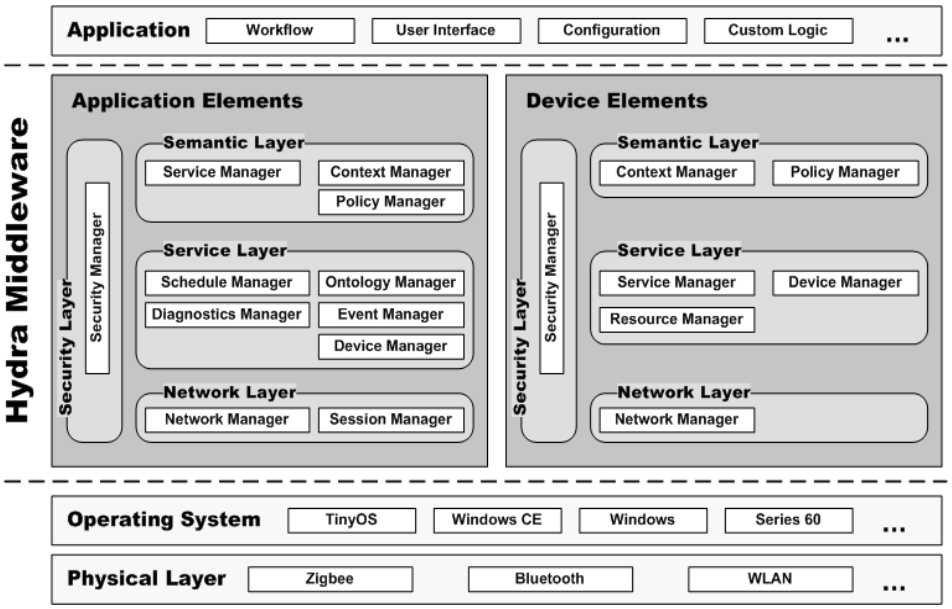

Fig. 2 - Structural overview of the middleware layers.

The HYDRA middleware elements are enclosed by the physical and the application layer shown at the bottom and at the top of the diagram respectively. The physical layer realizes several network connections like Zigbee, Bluetooth or WLAN. The application layer contains user applications which could contain modules like workflow management, user interface, custom logic and configuration details. These two layers are not part of the HYDRA middleware. The middleware itself consists of three layers - the network, service and semantic layer. Each layer holds elements according to their functionality and purpose. Note, that some device elements have similar, and thus like-wise named, counterparts among the application elements. Both, device and application elements, have a Security Manager. To express, that this manager is an orthogonal service, it is depicted vertical and covers all three middleware layers.

6. Ontologies in HYDRA

6.1 Device ontology and Application Ontology Manager

One of the key components in the Hydra middleware is the Device Ontology, where all meta-information and knowledge about devices and device types are stored. Our Device Ontology is based on the FIPA Device Ontology [4], which specifies a frame-based structure to describe devices, and is intended to facilitate agent communication for purposes such as content adaptation. Device description contains basic information related to a device such as the device name, vendor details, hardware description and software description used to describe hardware and software resources of the device. The purpose of the Application Ontology Manager is to provide an interface for using the Device Ontology. This manager could possibly also maintain other models in addition to devices.

Main Functions:

- Device description & annotation
- Parsing & annotation of device description
- Search/Query function
- Update
- Ontology versioning
- Reasoner module

Main Components:

- **Reasoner:** The reasoner module is responsible for reasoning about devices and their status and provides inferencing mechanisms for instance to conclude what type of device has entered the network.
- **Query module:** The query module allows for retrieving information regarding devices and their capabilities.
- **Update module:** The update module allows entering of new information, deletion and changes to the ontology at both design time and run time.
- **Versioning:** The versioning module is responsible for managing different version of the ontology. This includes different versions of devices and services.
- **Parse & Annotate:** The parse & annotate modules is responsible for automatically update the ontology with new device types. It does so by analyzing and annotates existing device and product descriptions which are fed into the ontology.

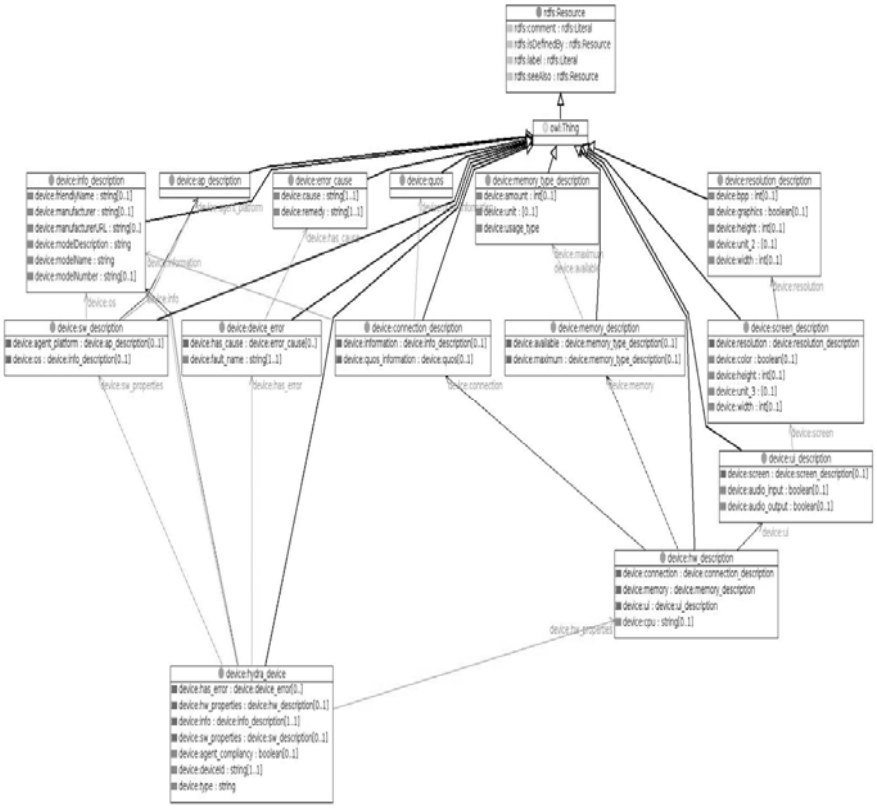

Fig. 3. Device Ontology in HYDRA

The semantic description will, in a general way, define the capabilities of the devices and the security requirements of the device. This is necessary in order to support interoperability among heterogeneous and distributed Hydra-enabled devices, but it also facilitates the task of virtualisation. The semantic description should allow individual devices to specify the type and level of security services that they both require and provide. This allows Hydra to match devices that may operate in concert, i.e., where the security requirements of all the devices are satisfied, and to reason about the overall level of security achieved by the aggregation of these devices. For example, messages delivered through different devices (e.g., in multi hop-/mesh networks) will be forwarded by devices that provide different levels of security, so it is important to be able to determine the minimum level of security offered by these devices. Without knowing the security services provided by the different devices, it will be impossible to reason about the overall security that can be guaranteed for such communication. Semantic parameterisation of devices according to their capabilities and their associated security domains will help in finding an optimum solution to these types of situations and hence it is vital to Hydra that we define clear semantic

framework representation for interoperability of devices to support secure, trusted services. This is essential for heterogeneous device networks where devices might exist which have very few features in common.

6.2 Semantic Description of Security

A semantic description of security should identify the elements of the security services which provide the actual security and define the type/level of security guaranteed by that service. This is probably best illustrated by an example. Consider a security mechanism based on the accountability of principals, e.g., a typical ACL-based mechanism used in a company that may dismiss an employee who misbehaves. In this case, the security is provided by the identification and authentication mechanisms, but also by the accountability framework defined by the employment contract and the legal framework in force at the place of work. Including employment contracts, labour laws and possibly civil liability legislation into the semantic description is obviously excessive and it will, furthermore, hamper interoperability by making it difficult to match security requirements, so simpler ways to describe the semantics of security requirements, such as identification, authentication and accountability, are needed.

The device ontology is considered as one of the model components in the Hydra. This ontology with related semantic services will be used both in design- and in run-time. At design time (by users of the IDE/SDK) by allowing developers to query on device properties and functions, in run-time the ontology will be used by the various services for device management (discovery, updates etc.).

We foresee that a basic ontology support is part of the generic Hydra platform, with extension possibilities given to developers. A user (of the IDE/SDK) in the future should be able to manually update device ontology. The system should also provide a certain level of automated updates, by generating and updating ontology contents from device/product documentations. Similarly, detection of modifications made to a device (e.g., a vendor software upgrade) should be possible and result in updates of corresponding ontology elements.

7. Conclusions

The software architecture presented in this paper is based on an iterative approach for the software architecture design process and the version presented in this paper is only the first iteration. In order to design an adequate architecture it is absolutely necessary that one gathers high-quality requirements from stakeholders. We have chosen to analyze only the Building Automation domain in this first iteration of the HYDRA architecture design.

The architecture presented here will be the basis for the first prototype. In the next iteration we will continue to expand the prototype in terms of adding more viewpoints and refining the requirements with input from the other domains.

Acknowledgements

The work presented in the paper is supported by the EC within the FP6 IST-2005-034891 Project "HYDRA - Networked Embedded System Middleware for Heterogeneous Physical Devices in a Distributed Architecture".

References

1. Hydra consortium: D2.1 Scenarios for usage of HYDRA in 3 different domains. Hydra Project Deliverable, IST project 2005-034891 (2006)
2. Hydra consortium: D2.5 Initial Requirements Report. Hydra Project Deliverable, IST project 2005-034891 (2006)
3. Sarnovsky, M., Butka, P., Kostelnik, P., Lackova, D.: HYDRA – Network Embedded System Middleware for Ambient Intelligent Devices, In: ICCC'2007: Proceedings of 8th International Carpathian Control Conference, Strbska Pleso, Slovak Republic, May 24-27 (2007) 611-614
4. FIPA. FIPA Specifications. http://www.fipa.org/specs/fipa00091/
5. Karnouskos, S., Baecker, O., de Souza, L. M., Spiess, P.: Integration of SOA-ready networked embedded devices in enterprise systems via a cross-layered Web Service infrastructure, 12th IEEE Conference on Emerging Technologies and Factory Automation (ETFA 2007), held in Patras, Greece, from 25 to 28 September 2007.