Smart Semantic Middleware for the Internet of Things

Abstract. As ubiquitous systems become increasingly complex, traditional solutions to manage and control them reach their limits and pose a need for self-manageability. Also, heterogeneity of the ubiquitous components, standards, data formats, etc, creates significant obstacles for interoperability in such complex systems. The promising technologies to tackle these problems are the Semantic technologies, for interoperability, and the Agent technologies for management of complex systems. This paper describes our vision of a middleware for the Internet of Things, which will allow creation of self-managed complex systems, in particular industrial ones, consisting of distributed and heterogeneous components of different nature. We also present an analysis of issues to be resolved to realize such a middleware.

1 Introduction

Recent advances in networking, sensor and RFID technologies allow connecting various physical world objects to the IT infrastructure, which could, ultimately, enable realization of the Internet of Things and the Ubiquitous Computing visions. Also, this opens new horizons for industrial automation, i.e. automated monitoring, control, maintenance planning, etc, of industrial resources and processes. A much larger, than in present, number of resources (machines, infrastructure elements, materials, products) can get connected to the IT systems, thus be automatically monitored and potentially controlled. Such development will also necessarily create demand for a much wider integration with various external resources, such as data storages, information services, and algorithms, which can be found in other units of the same organization, in other organizations, or on the Internet.

The interconnectivity of computing and physical systems could, however, become ”the nightmare of ubiquitous computing” [1] in which human operators will be unable to manage the complexity of interactions in the system, neither even architects will be able to anticipate that complexity, and thus to design the system. It is widely acknowledged that as the networks, systems and services of modern IT and communication infrastructures become increasingly complex, traditional solutions to manage and control them seem to have reached their limits. The IBM vision of autonomic computing (e.g. [1]) proclaims the need for computing systems capable of ”running themselves” with minimal human management which would be mainly limited to definition of some higher-level policies rather than direct administration. The computing systems will therefore be self-managed, which, according to the IBM vision, includes self-configuration, self-optimization, self-protection, and self-healing.

The vision of autonomic computing emphasizes that the run-time self-manageability of a complex system requires its components to be to a certain degree autonomous themselves. Following this, we envision that the *software agent technologies* will play

an important part in building such complex systems. Agent-based approach to software engineering is also considered to be facilitating the design of complex systems [2].

A major problem is inherent heterogeneity in ubiquitous computing systems, with respect to the nature of components, standards, data formats, protocols, etc, which creates significant obstacles for interoperability among the components of such systems. *Semantic technologies* are viewed today as a key technology to resolve the problems of interoperability and integration within heterogeneous world of ubiquitously interconnected objects and systems. Semantic technologies are qualitatively stronger approach to interoperability than contemporary standards-based approaches [3]. The Internet of Things should become in fact the *Semantic Web of Things* [4]. We subscribe to this view. Moreover, we apply semantic technologies not only to facilitate the discovery of heterogeneous components and data integration, but also for the behavioral control and coordination of those components (i.e. prescriptive specification of the expected behaviour, declarative semantic programming).

It seems to be generally recognized that achieving the interoperability by imposing some rigid standards and making everyone comply could not be a case in open ubiquitous environments. Therefore, the interoperability requires existence of some middleware to act as the glue joining heterogeneous components together. A consistent set of middleware, offering application programming interfaces, communications and other services to applications, will simplify the creation of services and applications and help to move from static programming approaches towards a configurable and dynamic composition capability [5].

There are a couple of EU FP6 research projects that have as one of their goals the development of some middleware for embedded systems. They are RUNES (Reconfigurable Ubiquitous Networked Embedded Systems, 2004-2007) and ongoing SOCRADES (Service-Oriented Cross-Layer Infrastructure for Distributed Smart Embedded Devices, 2006-2009). We believe, however, that the middleware needs of the Internet of Things domain go well beyond interconnectivity of embedded systems themselves. There is a more general need for middleware to enable something we refer to as Global Enterprise Resource Integration (GERI), where all different types of resources get seamlessly integrated: physical devices with embedded electronics, web services, software applications, humans along with their interfaces, and other. In the concept of GERI, we also stress the need for *true global interoperability, not just interconnectivity*. The components of ubiquitous computing systems should be able not only to communicate and exchange data, but also to flexibly coordinate with each other, discover and use each other, and jointly engage in different business processes.

Such more general middleware needs are emphasized in the Strategic Research Agenda (SRA) of the ARTEMIS European Technology Platform[1]. ARTEMIS' SRA has "Seamless Connectivity and Middleware" as one of its three parts. Some of the relevant research priorities listed are the middleware as the key enabler for *declarative programming* paradigm, where the components and their interaction are defined and configured declaratively rather than programmatically (and we believe that the semantic technologies are a natural candidate here), efficiently bridging information between

[1] ARTEMIS – Advanced Research and Technology for Embedded Intelligence and Systems: http://www.artemis-office.org

global, enterprise, and embedded systems, use of *ontologies* for cross-domain systems' organization and for interoperability in heterogeneous environments, dynamic reconfiguration capabilities, adaptive resource management, and appropriate security infrastructures.

In this paper, we describe our vision of such a middleware for the Internet of Things, which has also formed the basis for the recently-started research project MIDDLEWARE[2] (2007-2010). The project aims at a new generation middleware platform (MIDDLEWARE) which will allow creation of self-managed complex systems, in particular industrial ones, consisting of distributed, heterogeneous, shared and reusable components of *different* nature, e.g. smart machines and devices, sensors, actuators, RFIDs, web-services, software components and applications, humans along with their interfaces, and others. Such middleware will enable various components to automatically discover each other and to configure a system with complex functionality based on the atomic functionalities of the components.

The rest of the paper is structured as follows. Section 2 describes our general MIDDLEWARE vision. Section 3 presents our initial analysis of how the goals of MIDDLEWARE can be achieved. The result is a set of sub-problems, which we address in corresponding work-packages of the project. Section 4 describes several industrial cases (application areas) that we consider in the project; those are proposed by the project's industry partners. Finally, Section 5 concludes the paper.

2 The General Vision

This section describes our general vision of the smart semantic middleware for the Internet of Things.

We believe that tasks of automatic integration, orchestration and composition of complex systems on the Internet of Things will be impossible in a centralized manner due to the scalability issue. Therefore, the components of such systems should be to a certain degree autonomous and proactive. In other words, utilization of the agent technologies is needed to enable flexible communication and coordination of the components. Agent technologies also allow mobility of service components between various platforms, decentralized service discovery, utilization of FIPA communication protocols, and negotiation-based integration/composition of services. Interoperability among the components requires use of metadata and ontologies. As the amount of components can grow dramatically, without their ontological classification and (semi- or fully-automated) semantic annotation processes, the automatic discovery will be impossible.

Figure 1 depicts, in a very general way, our vision. In this vision, each resource to be connected has a representative – an autonomous software agent (a proactive "player" within certain integration scenario). This agent is assumed to be able to monitor the state of the resource, make decisions on the behalf on the resource, and to discover, request and utilize external help if needed. The connection between the resource itself and its agent is organized through the semantic mediation of an *adapter* (or interface). Such an adapter may include (if necessary) sensors with digital output, data structuring

[2] The name of the project as well as the name of the result system is changed to facilitate blind review.

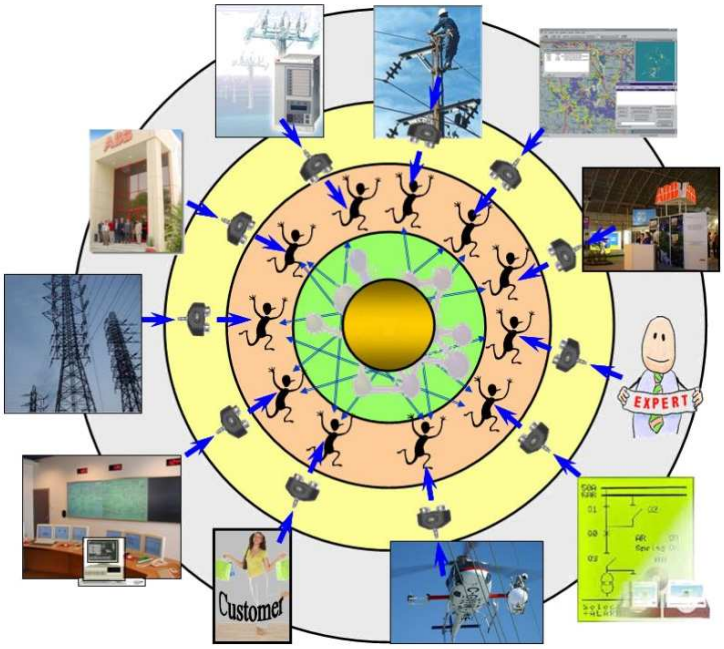

Fig. 1. The conceptual architecture

(e.g. XML), semantic adapter components (converting to a semantic representation), and actuators.

According to this vision, each relevant component of a ubiquitous system is to become a *"smart resource"* - proactive and self-managing. The part-of hierarchy of resources may result in corresponding hierarchy of the agents (Figure 2). This can also be recursive. For example, an adapter of a system component can become a smart resource itself, i.e. it can have its own responsible agent, semantically adapted sensors and actuators, history, commitments with other resources, and self-monitoring, self-diagnostics and self-maintenance activities. This could be done to guarantee high level of dynamism and flexibility of the adapter.

The semantic technologies have in our vision a two-fold value. First, they are the basis for the discovery of heterogeneous resources and data integration across multiple domains (a well-known advantage). Second, they are used for behavioural control and coordination of the agents representing those resources (a novel use). Therefore, semantic technologies are used both for *descriptive* specification of the services delivered by the resources and for *prescriptive* specification of the expected behaviour of the resources as well as the integrated system. Such two-fold application of semantics can, in ideal case, lead to a "global understanding" between the resources. This means that a resource A can understand all of (1) the properties and the state of a resource B, (2) the potential and actual behaviors of B, and (3) the business processes in which A and B, and maybe other resources, are jointly involved.

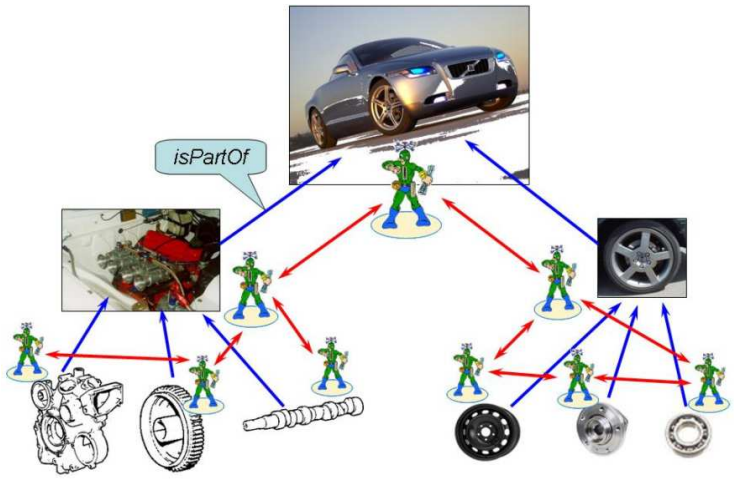

Fig. 2. Hierarchy of agents

The MIDDLEWARE is to be, roughly speaking, a set of tools providing means for effective development of agents and adapters, and a run-time environment for the operation of those. An important central component of MIDDLEWARE is the *agent core* (see Section 3.1). Each agent is assumed to be created based on this core and specialized through semantic declarative programs and a set of Reusable Atomic Behaviors (RAB). Combination of the run-time environment, the agent core and a set of standard semantic programs and RABs will make MIDDLEWARE a platform providing both (semantic) communication services and collaboration-support (scenario-driven integration) services for heterogeneous resources.

In one sense, our intention to apply the concepts of automatic discovery, selection, composition, orchestration, integration, invocation, execution monitoring, coordination, communication, negotiation, context awareness, etc (which were, so far, mostly related only to the Semantic Web-Services domain) to a more general ”Semantic Web of Things” domain. Also we want to expand this list by adding automatic self-management including (self-*)organization, diagnostics, forecasting, control, configuration, adaptation, tuning, maintenance, and learning.

3 Analysis

This section presents our initial analysis of how the goals of MIDDLEWARE can be achieved. The result is a set of sub-problems, which we address in corresponding work-packages of the project. MIDDLEWARE project aims at a relatively complete and self-sufficient middleware platform. Therefore, MIDDLEWARE elaborates on our central ideas (Section 2), and also works towards solutions in supporting but mandatory areas such as security, human interfaces and other.

3.1 Core Agent Platform

In the MIDDLEWARE vision, every resource, i.e. a component of a ubiquitous computing system, is represented by a software agent. Although the flexibility of agent interactions has many advantages when it comes to engineering complex systems, the downside is that it leads to unpredictability in the run time system; as agents are autonomous, the patterns and the effects of their interactions are uncertain [6]. It is common in specific systems and applications to circumvent these difficulties by using interaction protocols whose properties can be formally analyzed, by adopting rigid and preset organizational structures, and/or by limiting the nature and the scope of the agent interplay. However, these restrictions also limit the power of the agent-based approach; thus, in order to realize its full potential some longer term solutions are required [6].

Realization of the MIDDLEWARE vision requires a reliable core platform that would provide means for building systems that are *flexible* and consist of autonomous components, yet *predictable* in operation. Two important research directions, acknowledged in the literature, are: social level characterization of agent-based systems, and ontological approaches to coordination. The former direction presents the need for a better understanding of the impact of sociality and organizational context on an individuals behavior and of the symbiotic link between the behavior of the individual agents and that of the overall system [6]. In particular, it requires modeling behavior of an agent as being defined or restricted by the roles the agent plays in one or several organizations [7]. The latter direction presents the need to enable agents to communicate their intentions with respect to future activities and resource utilization and to reason about the actions, plans, and knowledge of each other, in real time [8].

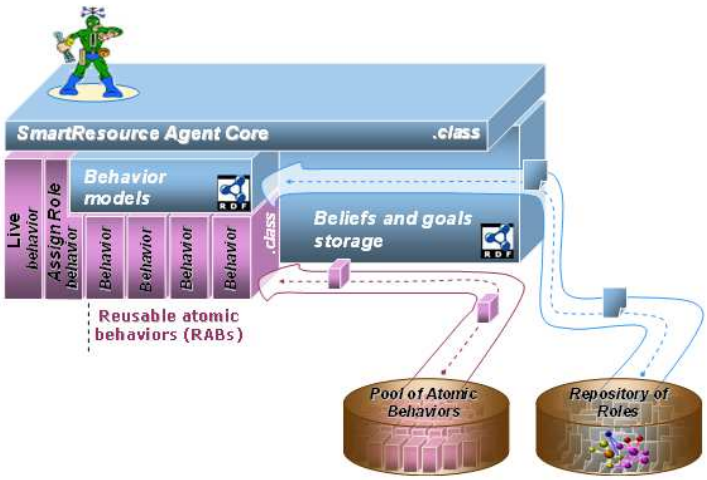

Fig. 3. Our current platform.

Our previous work has resulted in a platform [9] (Figure 3) that has done some steps into both these directions. It can be seen as consisting of three layers: reusable atomic behaviors (RAB), behavior models corresponding to different roles the agent plays, and the behavior engine (or the agent core). A RAB is a piece of Java code implementing a reasonably atomic function (this covers sensors, actuators, and data processing). A behavior model is an RDF-based document that is supposed to specify a certain organizational role. A behavior model consists of a set of beliefs representing the knowledge needed for playing the role and a set of behavior rules. Roughly speaking, a behavior rule specifies conditions of (and parameters for) execution of various RABs. The behavior engine is the same for all the agents (we mean that each agent has a copy of it). It is responsible for parsing RDF-based scripts, and it implements the run-time loop of an agent. In the platform, the agents access the behavior models from an external repository, which is assumed to be managed by the organization which "hires" the agents to enact those roles. As can be seen from the picture, the platform allows also on-demand access of RABs.

Such a 3-layer agent architecture with externalization of behavior models and on-demand access of atomic code components provides a good starting point for development of the MIDDLEWARE core platform. However, the present solution is very limited. The following important research questions have to be yet answered:

1. How the present language for roles' scripts has to evolve to enable the full spectrum of possibilities that is found in Agent Programming Languages (APLs) – to become, in addition to other benefits, a full-scale Semantic APL.
2. Is it important and, if yes, how to implement the separation between a role's capabilities (individual functionality), and the business processes in which this role can be involved (complex functionality)?
3. How to realize an agents roles as higher-level commitments of the agent that restrict its behavior, still leaving freedom for learning and adaptation on lower-levels, instead of totally and rigidly prescribing the behavior? (The latter is the case in the current platform, where the predictability was favored and the flexibility suppressed.)
4. What mechanisms are needed for flexibly treating the potential (and likely) conflicts among the roles played by one agent?
5. How to enable agents to flexibly discover each other, based both on the roles played and on particular capabilities possessed.
6. What would be concrete benefits of and what mechanisms are needed for accessing and using a role's script by agents who are not playing that role but wish to coordinate or interact with an agent that does?

Also, obviously, the core platform must and will provide support and enable the implementation of results from all the other work-packages of the project.

3.2 Managing Distributed Resource Histories

In the MIDDLEWARE vision, every resource is represented by a software agent. Among major responsibilities of such an agent is monitoring the condition of the resource and

the resources interactions with other components of the system and humans. The beliefs storage of the agent will, therefore, naturally include the history of the resource, in a sense "blogged" by the agent. Obviously, the value of such a resource history is not limited to that particular resource. A resource may benefit from the information collected with respect to other resources of the same (or similar) type, e.g. in a situation which it faces for the first time while other may have faced that situation before. Also, mining the data collected and integrated from many resources may result in discovery of some knowledge important at the level of the whole ubiquitous computing system.

A straightforward approach would be maintaining a central repository integrating the histories of all the resources of the system. However, such an approach is often impractical in realistic dynamic applications because even just keeping one agent informed of all the events and actions in the system would swamp the available bandwidth, and also such agent would become a severe bottleneck and might render the remaining components unusable if it failed [10]. A scalable solution requires mechanisms for inter-agent information sharing and data mining on integrated information which would allow keeping the resource histories distributed without need to copy those histories to a central repository.

The research questions, which are to be answered, are:

1. How to semantically markup the history of a resource in a system, in order to make it reusable for other resources and at the system-level?
2. What mechanisms are needed for effective and efficient sharing of information between the agents representing different resources?
3. How to query and to integrate responses from distributed, autonomous, and, hence, inevitably semantically heterogeneous resource histories?
4. How to perform mining (utilizing intelligent data mining and machine learning techniques) of distributed histories?

3.3 Peer-to-Peer Discovery

Following the IEEE FIPA agent system model, the MIDDLEWARE platform is to include a system agent called Directory Facilitator (DF). In MIDDLEWARE, the Directory Facilitator maintains a mapping between agents and the roles they play. If the behavior model of an agent X (see Section 3.1) prescribes the need of interaction with another agent Y, the agent Y is always specified by its role, not the name or another unique identifier of a particular agent. Therefore, the agent X must contact the Directory Facilitator in order to discover the unique ID (needed for communication) of the agent or agents playing the role needed. Also, because every agent plays at least one role, DF naturally has a list of all the agents on the platform. This can be used, e.g., when an agent needs a certain piece of information and wants to broadcast the request for that piece to all the agents on the platform (see Section 3.2).

Existence of such a Directory Facilitator is an effective and efficient solution. However, DF obviously presents a severe bottleneck in the system, and can render the whole system unusable if it failed. To improve the survivability of MIDDLEWARE-based systems, there has to be a complementary mechanism which can be utilized in an exception situation where DF became for some reason unavailable.

After the system is deployed and operational for some time, a significant part of the DF knowledge, piece by piece, ends up in local knowledge storages of different platform agents. The combination of the local storages presents thus a kind of distributed directory. Therefore, a Peer-to-Peer (P2P) mechanism implemented on such a distributed directory can work as the needed mechanism complementary to the central DF. Of course, replicating DF is another and probably simpler option. It would not, however, provide the same level of survivability as P2P. Also, in some business scenarios, it is possible that some of the services would prefer not to advertise themselves through the central DF altogether, for security or other reasons. Therefore, P2P discovery of such services would not be an exception path but the only viable solution.

The objective here is the design of mechanisms which will extend the scale of semantic resource discovery in UBIWARE with P2P discovery. Such mechanisms have to enable an agent:

- To discover agents playing a certain organizational role,
- To discover an agent (or agents) possessing certain needed information.
- To discover resources (through its agents) of certain type or possessing certain properties (e.g. a device in state X, a web-resource providing some information searched for, or a human with some specific skills).

In all cases, existence of a central DF is not assumed. Rather, the request is sent to all/some of the agents on the contact list of the agent in question (ones with who it has a history of communication, or at least about whom it heard from others). Those agents can forward the request to all/some of the agents on their lists, and so on.

3.4 Configurability

MIDDLEWARE aims to be a platform that can be applied in different application areas. This implies that the elements of the platform have to be adjustable, could be tuned or configured allowing the platform to run different business scenarios in different environments. Such flexibility calls for existence of a sophisticated configuration layer of the platform. All building blocks of the platform, i.e. software agents, agent behaviors, resource adapters, etc, become subject to configuration. On the other hand, a flexible system should have a long lifespan. Hence, the platform should allow extensions, component replacements, and component adjustments during the operation time.

In addition, every agent in MIDDLEWARE is self-aware and self-manageable entity. Therefore, it may evolve with time and modify some of its functional or non-functional properties. At the same time, we need a stable and predictable environment for running business scenarios. Thus, the adjustment made to a component must also be propagated to higher platform levels such as business processes and contracting. Also vice versa, a change in a business process may require some adjustments to be made to several participating components.

On the level of software design, we aim at defining patterns for development of configurable elements. On the level of inter-agent communication, we aim at establishing some protocols for negotiation in different configurability cases. One such case is tracing the dependencies when the configuration of one element changes. For example,

when an agent changes its behavior, a check must be performed of what are the consequences of this change for other components, and who and how should be informed about this change. Also, there should be a mechanism for restricting the range of possible adjustments to certain functionality of certain components in order to maintain the consistency in critical business processes.

One more issue is configurability of resource adapters. There is a need for a supportive and unified methodology both for creating such adapters and for their run-time re-configuration. It is reasonable to expect, e.g., that when a resource modifies the format of its output, the existing adapter for that resource can be adjusted to this new format without a need to build a new adapter from scratch.

Configuration of business processes poses a separate problem. A business process is an abstract entity which mainly consists of a flow of messages between agents. There is nothing like a business process execution engine. There could be a dedicated responsible agent, which would coordinate the involved agents, resolve conflicts and exceptions. However, similarly to the case of a centralized history storage (see Section 3.2), such a controller agent would become a severe bottleneck and might render the remaining components unusable if it failed. In a most scalable solution, a business process would be just a set of agents with some commitments, i.e. signed Service Level Agreements. In such autonomic medium, we will need a (re-)configuration mechanism for stable error-tolerant operation of components leading to successfully achieving the business goals.

3.5 Security in MIDDLEWARE

The security is often seen as an add-on feature of a system. However, in many systems (and MIDDLEWARE is one of them), the system remains nothing more but a research prototype, without a real potential of practical use, until an adequate security infrastructure is embedded into it.

MIDDLEWARE will advance existing technologies to a qualitatively new level and bring to life new complex ubiquitous environments, where traditional approaches to manage security fall short. Also, existing security measures for the technologies on which MIDDLEWARE relies, e.g. multi-agent, are not in a mature stage and still require significant elaboration to mitigate associated risks.

Traditional security goals like confidentiality, availability, reliability, integrity, manageability, accountability, responsibility etc, together with conventional measures and mechanisms that support security, do not cover all the needs and threats of new emerging computing environments. Ambient intelligence and ubiquity of information technologies have tighten the digital and physical worlds to the extent when security becomes the ultimate issue. The major implication of penetrating ICTs on security is that the risks and negative consequences of security threats become higher than ever. On the other hand, the security infrastructure itself has to become pervasive, interoperable and intelligent enough to naturally fit MIDDLEWARE. The security cannot be added to the MIDDLEWARE platform later but the design decisions regarding security have to be thoroughly correlated with the requirements, characteristics and design of the platform, due to mutual impact on resulting features of MIDDLEWARE.

The main objective here is the design of an infrastructure for policy-based optimal collecting, composing, configuring and provisioning of security measures in multi-agent systems like MIDDLEWARE [11]. This work is to follow the general MIDDLEWARE vision – configuring and adding new functionality to the underlying environment on-the-fly by changing high level declarative descriptions. Regarding security, this means that MIDDLEWARE will be able of smoothly including new, and reconfiguring existing, security policies (similar to semantic behaviour models) and mechanisms (similar to RABs), for the optimal and secure state of the MIDDLEWARE-based system, in response to the dynamically changing environment. The optimal state is always a tradeoff between security and other qualities like performance, functionality, usability, applicability and other.

3.6 Smart Interfaces

In MIDDLEWARE, humans are important resources, which can play several distinct roles:

- *User* – one getting some information or services from other resources.
- *Resource under care* – one under online care of the integrated system (e.g. monitoring the health of an employee)
- *Service provider* – one providing services to other resources (e.g. a maintenance expert).
- *MIDDLEWARE administrator* – one monitoring and configuring the integrated system.

Obviously, humans need some graphical interfaces to interact with the rest of the system. The same person can play several roles, switch between them depending on the context, and, in result, require different interfaces at different times. In addition, a MIDDLEWARE-based system presents a large integration environment with potentially huge amounts of heterogeneous data. Therefore, there is a need for tools facilitating information access and manipulation by humans.

From the MIDDLEWARE point of view, a human interface is just a special case of a resource adapter. We believe, however, that it is unreasonable to embed all the data acquisition, filtering and visualization logic into such an adapter. Instead, external services and application should be effectively utilized. Therefore, the intelligence of a smart interface will be a result of collaboration of multiple agents: the human's agent, the agents representing resources of interest (those to be monitored or/and controlled), and the agents of various visualization services. This approach makes human interfaces different from other resource adapters and indicates a need for devoted research. There is a need to enable creation of such smart human interfaces through flexible collaboration of an Intelligent GUI Shell, various visualization modules, which we refer to as MetaProvider-services, and the resources of interest [12].

A MetaProvider is responsible for acting as a portal so that various relevant resources can register on it, integration of data on the registered resources of interest, context-dependent filtering of data (including only relevant resources and relevant properties of those resources), creating visualization(s) of data. The GUI shell is, in turn, responsible for context-dependent selection of MetaProviders, communication with them, and cross-MetaProvider browsing and integration.

Based on such an approach, an infrastructure will be embedded into MIDDLEWARE enabling effective realization of the following system functions:

- Visualization of data provided by a service in response to a request
- Search, retrieving and visualization of data required by a human expert
- Providing access to contextual information, and visualization of it
- Visualization of resource registration, configuration, and security policy establishment processes
- Resource discovery via MetaProviders (because they act as thematic portals)

4 Industrial cases

This section describes several industrial cases (application areas) that we consider in our project. The objective of this line of work is to trial MIDDLEWARE on real industrial cases in a pursuit for two major goals. The first goal is to evaluate the scientific concepts behind MIDDLEWARE and to find problems and issues in MIDDLEWARE that would otherwise be overlooked. The second goal is to facilitate the further utilization of MIDDLEWARE in the industry. Several specific cases, proposed by the industrial partners, are to be analyzed, designed and prototyped based on the MIDDLEWARE platform. The reasons for prototyping are the same: to identify issues in MIDDLEWARE that would get overlooked if the work was only theoretical and thus abstract, and to demonstrate the benefits of MIDDLEWARE in a tangible way so to facilitate future industrial adoption.

As will be explained below, different industrial partners make accents on different benefits that UBIWARE provides. We see this as a positive moment, because the resulting prototypes will emphasize different MIDDLEWARE aspects and, together, will provide a nearly complete coverage of MIDDLEWARE functionality and issues.

4.1 Power Network Maintenance

One of the project's partners is a vendor of hardware and software for power networks. With respect to MIDDLEWARE, this partner's main area of interest is in extending, mainly through integration with external resources, the functionality of its existing software systems. These systems provide an integrated graphical view over the power network, provide data acquisition from the substations and remote control over the relays, switches, etc. They also include implementations of various algorithms: for fault localization, for calculation of optimal reconfiguration of the network and other.

The MIDDLEWARE technology could allow connecting to the existing products some new system intelligence tools, for example, statistical and data mining tools. One case that is considered is about possibility of using data mining techniques for automated interpretation of the situation in the power network. The motivating issue is that a certain condition, e.g. a fault, in the power network causes a series of alert messages to be sent to the operation center. It remains a responsibility of a human operator to understand the reason underlying such a sequence of alert messages. Based on MIDDLEWARE, a system could be created allowing that: (1) alert data is automatically

collected, (2) the experts are able to annotate sequences of events with their reasons, (3) data mining algorithms are applied to discover some patterns, and (4) the operator is provided with an interface giving a more structured view of events (e.g. filtered based on the source of event) and including the system's interpretation (from the models built) of what is the meaning of the situation and its reason.

In collaboration with this partner, we have analysed further the potential add-value which the company and their customers, i.e. electrical companies, could receive from introducing MIDDLEWARE into their businesses. Below, we sketch several scenarios when MIDDLEWARE can enable new features, or help otherwise.

One scenario is related to extending the user interfaces, so that various groups of users could get flexible access to data and functionality present in the existing products. Traditionally, those software products are only used inside the walls of operation centers by the network operators, through proprietary user interfaces. The data and functionality of those systems, however, has a value beyond that use. A MIDDLEWARE-based solution will enable a more ubiquitous and flexible information access to data and algorithms and will extend the user base to, e.g., maintenance workers, management, etc.

Another scenario arises from the fact that the medium-voltage sub-networks of the integral power network are usually owned, controlled and maintained then by some local companies. It is noticeable that the operation centers of different companies have no connection to each other, so information exchange among them is nearly impossible. In the case of a fault affecting two different sub-networks, such information exchange, though, may be very important, for all of fault localization, network reconfiguration, and network restoration. Introducing an inter-organizational system based on MIDDLEWARE could solve this issue. The information flow will go through the agents representing the sub-networks on the MIDDLEWARE platform. Utilization of Semantic technologies will allow such interoperability even if the sub-networks use software systems from different vendors, and thus maybe different data formats and protocols.

One more scenario is related to a new business model that could be implemented. At present, all expertise of the company gets embedded into hardware or software systems and sold to the customers as it is. A new business model would be to start own Web-service providing implementation of certain algorithms, so the customers will utilize those algorithms online when needed. The company will be always able to update algorithms, add new, and so on. MIDDLEWARE will ensure interoperability and coordination between such Web-service and customers' software systems, and also a relative ease of implementation of such a solution – because it will not require changes in existing software systems, only extension with the MIDDLEWARE platform. Noticeable that, if semantically defined, such Web-service can potentially be utilized across the globe even by the customers who never purchased any of the company's hardware or software.

The next scenario is related to the possibility of integrating data, which is currently utilized in the power network management (network structure and configuration, feeder relay readings), with contextual information from the external sources. Such integration can be used for:

- Risk analysis. Information about whether conditions, ongoing forest works, or forest fires can be used for evaluating existing threats for the power network. This

may be used to trigger an alert state for the maintenance team, or even to do a precautionary reconfiguration of the network to minimize possible damage.
- Facilitation of fault localization. The output of fault localization algorithms is not always certain. The information about threats for the power network that existed at the time when the fault occurred (which thus may have caused the fault) may greatly facilitate the localization. In some situations, contextual information alone may even be sufficient for localization.
- Operator interface enhancement. Contextual information may be used also just to extend the operators' view of the power network. For example, satellite imagery can be used for geographic view (instead of locally stored bitmaps as it is in the current systems); also, dynamically-changing information can be accessed and represented on the interface.

The last scenario is about the possibility of transferring the knowledge of human experts to automated systems, by means of various data mining tools. In the power network management case, one scenario that seems to be highly appropriate for such knowledge transfer is the following. In present, it is always a decision of a human expert which of the existing fault localization algorithms will perform the best in the context of the current configuration of the power network and the nature of the fault. Such decisions made by an expert along with the input data could, be forwarded to a learning Web-service. After a sufficient learning sample, this Web-service could start to be used in some situations instead of the human expert, e.g. in situations when a faster decision is needed or when the expert is unavailable.

4.2 Maintenance of Paper Machines

Another of the project's partners is a supplier of machinery and automation systems for a set of industries including the paper industry. With respect to MIDDLEWARE, this partner's main areas of interest are in information integration and analytical data processing. To support customers with additional information along the product (a paper machine) lifecycle, the company foresees the need for intelligent product history management. Such a history is supposed to integrate alert reports, experts' diagnoses, maintenance work performed, maintenance costs, and other types of information.

Every paper machine, installed at a customer's site, is equipped with a set of sensors and embedded intelligence for observing the state of the machine and alarming when an exceptional situation occurs. The alarm information is forwarded to the central hub for diagnostics and decision making.

MIDDLEWARE should provide means for integration of different corporate information systems in order to have a common basis for product history management and further intelligent data processing. MIDDLEWARE-based integration with external web services should enable more powerful analytical processing of data. Also, the flexibility and ease of extension via incorporation of external functionality will enable evolution of the service infrastructure to meet the customer needs in a sustainable fashion. The MIDDLEWARE platform will also enable a high level of security in the collaborative work of the company and its customers.

Among the central problems to be solved by MIDDLEWARE, there are those related to the evolution of the service infrastructure. During the paper machine operation

lifecycle, formats of messages or algorithms for issuing alarms may change. Furthermore, some changes to the supporting ERP or other automation systems may take place. In other words, data formats and data structures, which were bound to ontology, may require re-adaptation (i.e. adapter re-configuration) to become consistent with the integrated storage and applications. From a life-time perspective, the complexity of business processes between the company and the customer may become another obstacle for updates and renovation of the service infrastructure. It is because such changes are hard to trace and, therefore, it might be difficult to analyze their impact on the overall system.

4.3 Telecom Operator's Service Desk

One more project's partner is a provider of telecommunication services, both wireless and fixed-line. With respect to MIDDLEWARE, this partner's main area of interest is in possibility of constructing, mainly through integration of existing components, systems that would (partially) automate some traditionally manual processes. Such a manual process to be considered is a Service Desk. Among other things, the company's service desk is a point of contact for the customers seeking to report unavailability of service (mobile or fixed-line phone connection, IDSL Internet connection, and other) or another problem. If the problem is at the customer side, the service desk experts are supposed to provide instructions for troubleshooting it. The whole interaction is performed over the phone, i.e. the customer calls, explains the problem to a service desk worker, probably gets transferred to a relevant expert, tries what the expert recommends, reports the results, and so on. In result, the waiting times for customers to get through to the service desk are long, while the effectiveness of the troubleshooting process is quite low.

Equipping the service desk with an MIDDLEWARE-based solution could enable the following new functionality:

- Customers get possibility to report their problems through a web-service interface.
- The system could collect, integrate and represent to the expert all the available information on the customer (type of connection, address, etc), removing a need to spend time asking that information during the phone conversation.
- Automated analysis of related problem reports could enable fast responses like: "it is a network problem, we have received other similar reports from your area, we are already working on this problem".

In the case of a non-working Internet connection, some relevant data can be acquired, i.e. sensed, from the network side. Also, some modems can be configured remotely by uploading a configuration file from the network. In such a case, a fully automatic troubleshooting could even be implemented.

5 Conclusions

In this paper, we describe our vision of a solution to meet the middleware needs of the domain of the Internet of Things. We aim at a new generation middleware platform

(MIDDLEWARE) which will allow creation of self-managed complex systems, in particular industrial ones, consisting of distributed, heterogeneous, shared and reusable components of different nature.

Self-management of systems is one of the central themes in the EU 7-th Framework ICT Programme (2007-2013). The Objective "Service and Software Architectures" of the Challenge 1 "Network and Service Infrastructures" includes the need for strategies and technologies enabling mastery of complexity, dependability and behavioral stability, and also the need for integrated solutions supporting the networked enterprise. Also, the Objective "The Network of the Future" of this Challenge includes the need for reconfigurability, self-organization and self-management for optimized control, management and flexibility of the future network infrastructure. In addition, the whole Challenge 2 "Cognition, Interaction, Robotics" has as its motivation the need for creating "artificial systems that can achieve general goals in a largely unsupervised way, and persevere under adverse or uncertain conditions; adapt, within reasonable constraints, to changing service and performance requirements, without the need for external reprogramming, re-configuring, or re-adjusting". It is noticeable that the systems (standalone or networked) monitoring and controlling material or informational processes is one of the three focus areas of this Challenge.

According to a more global view to the Internet of Things technology, MIDDLWWARE will classify and register various ubiquitous devices and link them with web resources, services, software and humans as business processes' components. MIDDLEWARE will also consider sensors, sensor networks, embedded systems, alarm detectors, actuators, communication infrastructure, etc. as "smart objects" and will provide similar care to them as to other resources.

The innovative nature of MIDDLEWARE is demonstrated well by the very idea of the case in Section 4.3, the smart service desk. An operator's service desk is an important service element which, however, still remains largely non-automated and thus has a low effectiveness. Any automation of such a process would require integration and complex interoperability of highly heterogeneous components, including hardware (e.g. customer's equipment), software and database systems, and humans (the customer, the service desk operator, and experts). In result, the possibility of automation of the service desk was not really considered before. However, the introduction of the MIDDLEWARE concept has led to realizing that some automation can be possible after all.

References

1. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. IEEE Computer **36**, 1 (2003) 41–50
2. Jennings, N.: An agent-based approach for building complex software systems. Communications of the ACM **44**, 4 (2001) 35–41
3. Lassila, O.: Using the Semantic Web in mobile and ubiquitous computing. In: Proc. IFIP WG12.5 Conference on Industrial Applications of Semantic Web, Springer (2005) 19–25
4. Brock, D., Schuster, E.: On the semantic web of things. In: Semantic Days 2006, Stavanger, Norway, April 26-27. (2006)

5. Buckley, J.: From RFID to the Internet of Things: Pervasive networked systems. In: Report on the Conference organized by DG Information Society and Media, Networks and Communication Technologies Directorate, Brussels, March 6-7. (2006)
6. Jennings, N.: On agent-based software engineering. Artificial Intelligence **117**, 2 (2000) 277–296
7. Vazquez-Salceda, J., Dignum, V., Dignum, F.: Organizing multiagent systems. Autonomous Agents and Multi-Agent Systems **11**, 3 (2005) 307–360
8. Tamma, V., Aart, C., Moyaux, T., Paurobally, S., Lithgow-Smith, B., Wooldridge, M.: An ontological framework for dynamic coordination. In: Proc. 4th International Semantic Web Conference'05, LNCS vol. 3729, Springer (2005) 638–652
9. : Self-reference is hidden to facilitate blind review. (2007)
10. Jennings, N.: Commitments and conventions: The foundation of coordination in multiagent systems. Knowledge Engineering Review **8**, 3 (1993) 223–250
11. : Self-reference is hidden to facilitate blind review. (2007)
12. : Self-reference is hidden to facilitate blind review. (2007)