SEMANTIC WEB SERVICES FOR SMART DEVICES BASED ON MOBILE AGENTS

Vagan Terziyan

Department of Mathematical Information Technology, University of Jyvaskyla,
P.O. Box 35 (Agora), FIN-40014 Jyvaskyla, Finland
vagan@it.jyu.fi

ABSTRACT

Among traditional users of Web resources industry has also a growing set of smart industrial devices with embedded intelligence. As well as humans they need online services, e.g. for condition monitoring, remote diagnostics, maintenance, etc. In this paper we present one possible implementation framework for such Web services. Assumed that such services should be Semantic Web enabled and form a Service Network based on internal and external agents' platforms, which can host heterogeneous mobile agents and coordinate them to perform needed tasks. Concept of a "mobile service component" assumes not only exchanging queries and service responses but also delivering and composition of a service provider itself. Mobile service component carrier (agent) can move to a field device's local environment (embedded agent platform) and perform its activities locally. Service components improve their performance through online learning and communication with other components. Heterogeneous service components' discovery is based on semantic P2P search.

Keywords: Mobile Agents, Semantic Web, Web-Services Field Devices, Condition Monitoring, Maintenance.

1. INTRODUCTION

The intersection of Web Service, Semantic Web and Enterprise Integration Technologies are recently drawing enormous attention throughout academia and industry (Bussler *et al*, 2003) and the expectation is that Web Service Technology in conjunction with Semantic Web Services will make Enterprise Integration dynamically possible for various enterprises compared to the "traditional" technologies (Electronic Data Interchange or Value Added Networks).

The Semantic Web is an initiative of the World Wide Web Consortium with the goal of extending the current Web to facilitate Web automation, universally accessible content, and the "Web of Trust". Tim Berners-Lee (Berners-Lee *et al*, 2001) has a vision of a Semantic Web, which has machine-understandable semantics of information, and trillions of specialized reasoning services that provide support in automated task achievement based on the accessible information. Management of resources in Semantic Web is impossible without use of ontologies, which can be considered as high-level metadata about semantics of Web data and knowledge (Chandrasekaran *et al*, 1999). DAML-S or DAML for Services (Ankolekar *et al*, 2002; Paolucci *et al*, 2002) provides an upper ontology for describing properties and capabilities of Web services in an unambiguous, computer interpretable markup language, which enables automation of service use by agents and reasoning about service properties and capabilities. There is a also a growing interest in the use of ontologies in agent systems as a means to facilitate interoperability among diverse software components (Ontologies, 2003). The problems related to that are being highlighted by a number of recent large-scale initiatives (e.g. Agentcities, Grid computing, the Semantic Web and Web Services). A common trend across these initiatives is the growing need to support the synergy between ontology and agent technology.

The key to Web Services is on-the-fly software composition through the use of loosely coupled, reusable software components (Fensel *et al*, 2002). Still, more work needs to be done before the Web service infrastructure can make this vision come true. Among most important European efforts in this area one can mention the SWWS (Semantic Web and Web Services, swws.semanticweb.org) project, which is intended to provide a comprehensive Web Service description, discovery and mediation framework.

Usually a Web Service is accessed by human users or by applications on behalf of human users. However there already exists and growing a new group of Web Service "users", which are smart industrial devices, robots or any other objects equipped by "embedded intelligence" There is a need to launch special Web Services for such smart industrial devices. Such services will provide necessary

online information provisioning for the devices, allow the heterogeneous devices to communicate and exchange data and knowledge with each other and even support co-operation between different devices. There are quite many open questions to be answered within this research area.

In this paper we are trying to discuss the way of implementing emerging Semantic Web and Web services technologies to a real industrial domain, which is field device management. The goal of this paper is to discuss possible implementation framework to Web services that automatically follow up and predict the performance and maintenance needs of field devices.

The rest of the paper organized as follows. Chapter 2 briefly introduces our concepts of an intelligent agent and mobility. Chapter 3 presents two alternative architectures for distributed problem solving based on mobile agents. Chapter 4 describes the domain of field device maintenance and ways of implementing agents in it. Chapter 5 discusses implementation issues related to the Web service network (OntoServ.Net) of smart devices based on integration of Semantic Web services' and multiagent technologies. Chapter 6 concludes.

2. AGENTS, SEMANTIC BALANCE AND MOBILITY

In spite of existence of so many definitions for the concept of an intelligent agent we will use our own one. The definition will base on the concept of Semantic Balance (Terziyan & Puuronen, 1999). In Figure 1 the concept of internal and external environments is illustrated.

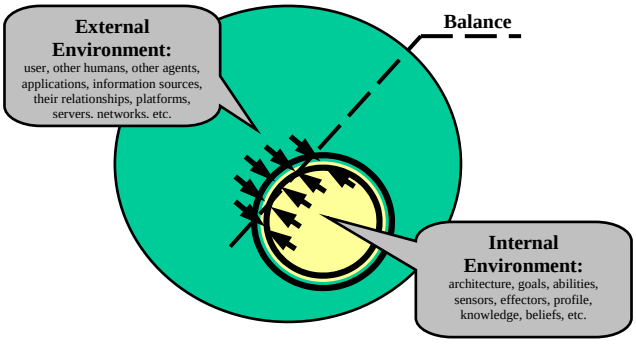

Figure 1. Internal and external environments of an agent

We consider Intelligent Agent as an entity that is able to *keep continuously balance between its internal and external environments* in such a way that in the case of unbalance agent can choose the behavioral option from the following list:

- *make a change within external environment* to be in balance with the internal one;
- *make a change within internal environment* to be in balance with the external one;
- find out and *move to another place* within the external environment where balance occurs without any changes;
- *communicate* with one or more other agents (human or artificial) to be able *to create a community*, which internal environment will be able to be in balance with the external one.

The above means that an agent:

1) is **goal-oriented**, because it should have at least one goal - to *keep continuously balance between its internal and external environments* ;
2) is **creative** because of the ability to *change external environment*;
3) is **adaptive** because of the ability to *change internal environment*;
4) is **mobile** because of the ability to *move to another place*;
5) is **social** because of the ability to *communicate to create a community.*

Thus we see the mobility is an important adaptation ability of an intelligent agent.

3. "MOBILE AND DISTRIBUTED BRAINS" ARCHITECTURES

Assume that there is certain intelligent task (e.g. remote diagnostics of a device based on sensor data), which appears somewhere in the Web. Assume also that necessary intelligent components ("distributed brains") to perform this task are distributed over the Web, e.g. in a form of Web-Services. Assume finally that there is also an intelligent engine able to perform integration of autonomous components for solving complex tasks.

Consider following two architectures for this distributed problem solving.

Mobile Engine architecture. To integrate distributed service components into one transaction to solve the task, the intelligent engine (e.g. mobile transaction management agent) makes necessary visits to all distributed platforms, which host these services, and provides all necessary choreography. Mobility here is an option, which can be replaced by remote access to the components.

Mobile Components architecture. Alternatively the necessary components discovered for performing the task move to the platform where engine is resized and choreography is performed locally. According to business

models around the concept of a Web-service, it is naturally to assume that services (intelligent components in our case) should be "self-interested" and whenever they move they should serve according the interests of their creators. This means that very appropriate concept for such components is the concept of mobile agents. Agent is self-interested entity, which can act according to certain goals whenever it appears.

Both architectures can be considered as appropriate for implementation of the environment for distributed condition monitoring and remote diagnostics Web-services for field devices, which is discussed in the following chapters of this paper.

4. FIELD DEVICE MANAGEMENT AND AGENT TECHNOLOGIES

The expectations from smart field devices include advanced diagnostics and predictive maintenance capabilities. The concerns in this area are to develop a diagnostics system that automatically follows up the performance and maintenance needs of field devices offering also easy access to this information. The emerging agent and communication technologies give new possibilities also in this field. Field device management in general consists of many areas of which the most important are:

- Selection
- Configuration
- Condition monitoring
- Maintenance

Valuable information is created during each phase of device management and it would be beneficial to save it into single database. This information can be utilized in many ways during the lifetime of the devices, especially as life cycle cost (or lifetime cost) of all assets is getting nowadays more and more attention. Accordingly the concept of life cycle management of assets has become very popular (Pyötsiä & Cederlöf, 1999).

Field Agent is a software component that automatically follows the "health" of field devices. This agent can be either embedded to a device (Lawrence, 2003) or resized at the local network. It is autonomous, it communicates with its environment and other Field Agents, and it is capable of learning new things and delivering new information to other Field Agents. It delivers reports and alarms to the user by means of existing and well-known technologies such as intranet and e-mail messages. Field device performance has a strong influence on process performance and reliable operation in more distributed process automation architecture based on FieldBus communication (Metso, 2003; Sensodec, 2003). In this situation, easy on-line access to the knowledge describing field device performance and maintenance needs is crucial. There is also growing need to provide automatic access to this knowledge not only to humans but also to other devices, applications, expert systems, agents etc., which can use this knowledge for different purposes of further device diagnostics and maintenance. Also the reuse of collected and shared knowledge is important for other field agents to manage maintenance in similar cases.

While monitoring field device via one information channel (Figure 2) one can get useful information about some dimension of the device state, then derive online some useful patterns from this information, which can be considered as "symptoms" of the device "health", and finally recognize these symptoms using "Ontology of Patterns".

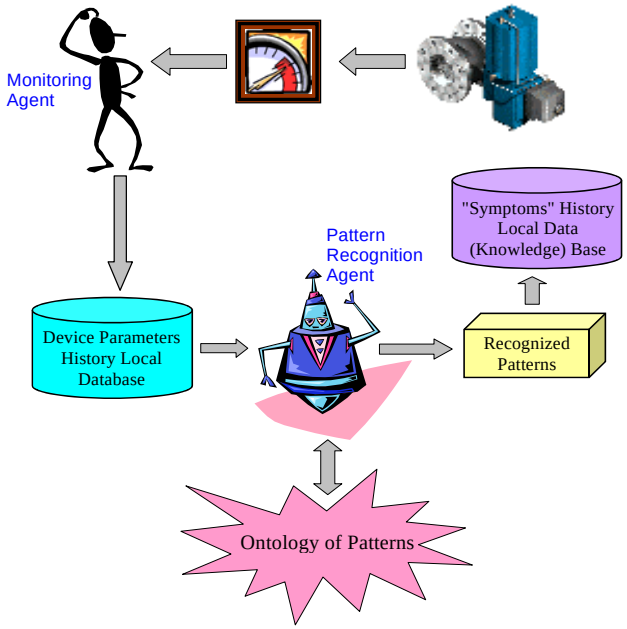

Figure 2. Agent-based symptom recognition in device monitoring

If to monitor a device via several information channels (Figure 3) then appropriate Field Agent Infrastructure allows not only deriving and recognizing "symptoms" of the device "health", but also deriving and recognizing a disease itself using "Ontology of Diseases". In any case history data, derived patterns and diagnoses can be stored and used locally however there should be a possibility to easy access this information and also to share it with other agents for reuse purposes.

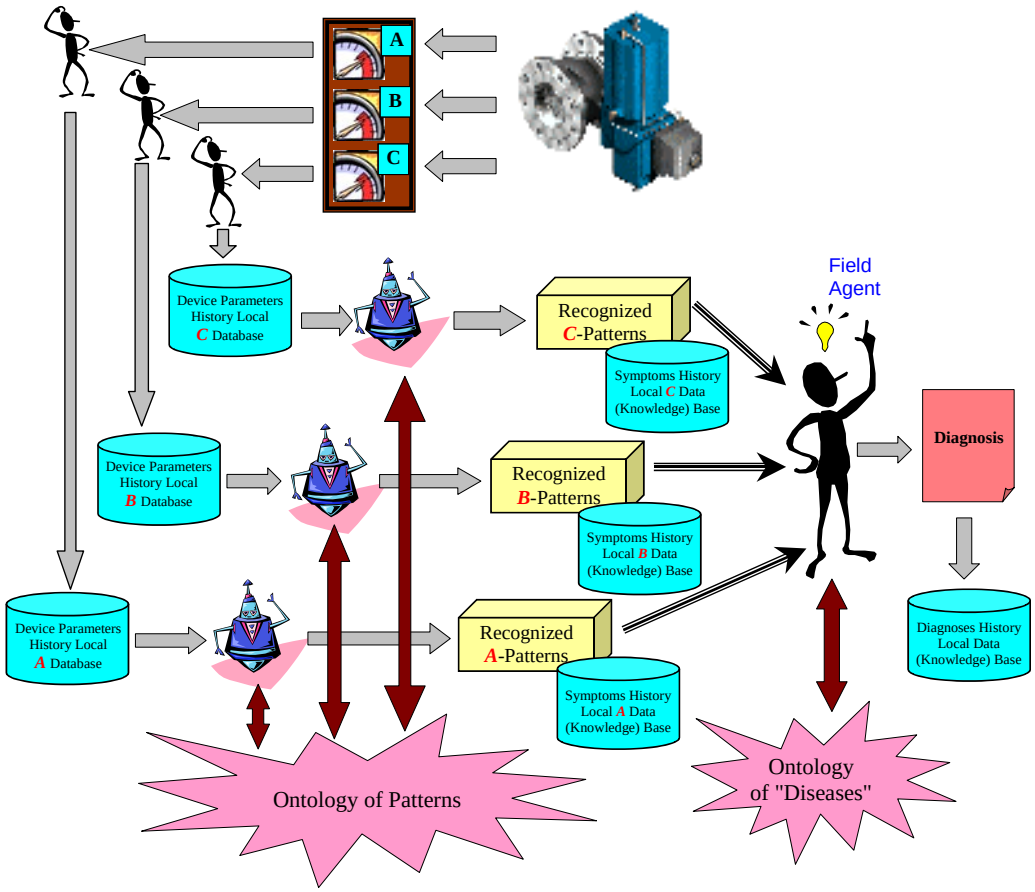

Figure 3. Agent-based diagnostics of field devices

There are at least two cases when such distributed infrastructure is reasonable. The first one is when we are monitoring a group of distributed devices, which are physically and logically disjoint, however they all are of the same type. In this case any history of derived patterns and diagnoses from one device can be useful to better interpret current state of any other device from the group.

The second case relates to the monitoring of a group of distributed devices of a different type, which are considered as a system of physically or logically interacting components. In such case it would be extremely important for every field agent to use outcomes from other field agents as a context for interpretation of the produced diagnosis. Thus in these two cases appropriate field agents should communicate with each other (e.g. in peer-to-peer manner) to share locally stored online and historical information and thus to improve the performance of the diagnostic algorithms, allowing even the co-operative use of heterogeneous field devices produced by different companies, which share common communication standards and ontologies.

We are considering case when (predictive) maintenance activities can be performed not only by humans but also by embedded automatics controlled by agents. We also assume that newest Semantic Web and Intelligent Web Services concepts can be applied to the problems of interoperability among field devices and will result to essential improvement of field device maintenance performance.

5. ONTOSERV.NET IMPLEMENTATION ISSUES

The OntoServ.Net concept was developed by Industrial Ontologies Group (http://www.cs.jyu.fi/OntoGroup) as a large-scale automated industrial environment for assets management. First of all, we consider maintenance of

assets, but, in general, this concept can be applied for process control, improvement of operating efficiency, field-performance diagnostics, etc., as well. Better maintenance provided by OntoServ.Net considers maintenance information integration, better availability of operational data and shift from reactive and preventive maintenance towards predictive and proactive maintenance, which means, first of all, reduced Total Life Cycle Cost of machines. OntoServ.Net is also a network of industrial partners, which can share maintenance methods and information developed during work of separate machine (device, equipment, installation). Improved locally, maintenance experience can be shared.

Also, it is assumed that there are special commercial maintenance (diagnostics) services supported either by manufactures of machines, or by third parties. Browsing a devices internal state is extended to an automatic diagnostics and recovery within a network of maintenance services or even within network of platforms hosting several maintenance services. The role of a maintenance service, firstly, is to organize gathering and integration of field data to learn based on it, and secondly, support its "clients" (field devices) providing remote diagnostics and maintenance services. Implementation of such large-scale environment as OntoServ.Net presents many problems to be solved. The challenge here is standardization of maintenance data across various software systems within OntoServ.Net and existing industrial systems.

5.1. Ontology-Based Standardization of Maintenance Data

We are focusing on maintenance data, which comes from field devices. For remote diagnostics (in case of predictive maintenance, for instance) these data needs to be sent to some other place beyond local computing system (whether it a computer or embedded system). We assume that maintenance network without centralized control requires some global standard for data representation in order to provide compatibility of network nodes.

We consider standardization based on ontological description of data. Ontology-based approach here stands as an alternative for development of maintenance-specific set of standards/vocabularies/procedures for information exchange. We use ontology concept and data representation framework, which was developed within Semantic Web activities. Ontology-based information management is going to be more flexible and scalable, and also it has potential to become next-generation standard of information exchange in the Internet. Ontology engineering phase includes development of upper-ontology (schema for ontology) and development of ontology itself, which includes specific data about maintenance domain (such as device descriptions, diagnostic methods descriptions, etc.) Concrete data is annotated in terms of upper- and common ontology. Here, ontology provides a basis for a well-understood "common language" to be used between devices and systems.

If to consider field devices as data sources, then information to be annotated is sensors' data, control parameters and other data that presents relevant state of the device for the maintenance process. Special piece of device-specific software (*OntoAdapter*) is used for translation of *raw diagnostic data* into *standardized maintenance data.* This adapter can be integrated into legacy system used for device management or can be developed independently from existing software if such solution is appropriate and possible.

Type of software, which uses data being described in an ontological way, can vary depending on needs. It can be a data browser, control panel of operator, computing system, database storage, etc. Because of the way data is represented, it will be never processed incorrectly, since software can check itself whether data semantics, as annotated, is the same or compatible, as data processing unit needs.

Additional benefit comes from data annotation for software development even if there is no need to deliver information outside of origin computing system: no more needs to develop special formats of maintenance data exchange between application, since it is already presented in common standard by means of ontology. Software can be developed in modular, scalable manner with support of this standard (ontology). Such commitment to the shared (upper-) ontology will provide compatibility of software.

5.2. Ontology-Based Diagnostics based on Maintenance Data

It is assumed that there are special commercial diagnostic units (maintenance services) supported either by manufactures of machines, or by third parties. Browsing a devices' internal state is extended to an automatic diagnostics and recovery within a network of maintenance services. As it was already mentioned, the role of maintenance service, firstly, is to organize gathering and integration of field data and learning based on it, and secondly, to support its "clients" (field devices) providing remote diagnostics services.

Considering aspects of maintenance network development, following statements are true:

1) There are diagnostic software components (further also mentioned as *classifiers* or *diagnostic units*), which perform predictive/proactive diagnostics. These diagnostic units obtain maintenance data delivered to them either locally, or from remote source, and provide diagnosis as an output.
2) Diagnosis provided by classifier can be of several types: which class of state an observed device has, what kind of in-depth diagnostics is and what [maintenance] actions/activities are required.
3) Diagnostic units are specialized in certain aspects of maintenance diagnostics, so device usually needs support from a set of different diagnostic units, which operate and can be replaced independently.
4) Diagnostic units (components), in general, are not device-specific and perform similar diagnostic tasks in variety of device monitoring systems.
5) Diagnostic units are [OntoServ.Net] platform-compatible and developed separately from maintained devices; thus, they can be used on any platform within OntoServ.Net.
6) Once diagnostic unit possesses ability to learn, the maintenance experience it got will be available for diagnostics of other devices. It is done by means of running copy of classifier on other maintenance platforms or presenting its experience in a way that will allow reusing it by other diagnostic units. Especial interest is in presented capabilities of integration of such information obtained "worldwide" and applied effectively for individual devices.
7) Maintenance platform is a computing environment in which maintenance services (diagnostic units etc.) are installed. It supports device-specific interfaces for connecting devices, on one side, maintenance managing core with installed maintenance services and, on other side, supports connection to a maintenance network consisting of maintenance platforms of other devices and maintenance platform of *maintenance services* – specialized centres for remote maintenance. Such services can be implemented based on agent technology, see e.g. (Gibbins *et al*, 2003). Taking into account the openness of the system the issues of security and trust management are considered as exceptionally important (Kagal *et al*, 2001);
8) Maintenance platform manages maintenance information exchange between devices and diagnostic units resided locally and elsewhere in the maintenance network. It also supports search of required network resources, upgrades of installed software, supports mobility feature for maintenance services.

Since available set of classifiers can vary, type of the classifiers (their purpose) is specified in order to allow their selection in cases, when there is a necessity. Every classifier has its description attached – information concerning its capabilities. Description also contains information how to use classifier, what inputs it requires and what output it provides.

Preliminary classification mechanism of a maintenance platform takes into accounts, which services (classifiers) are available now, and selects those, which declare themselves as able to deal with class of problems derived on pre-classification phase. Here, pre-classification is alike to human-operator work, who can detect some abnormal device behaviour and use analysis tools in order to find out the source of the problem.

It is supposed that some historical maintenance data is available and it is used for automatic learning what kind of maintenance actions should be performed. Learnt knowledge in this model supports rather simple, but automated reasoning mechanism, which implements preliminary diagnostics of device state and can identify certain deviations from normal operational state and run appropriate diagnostic service.

Industrial Ontologies Group proposes to involve into maintenance data processing the descriptions of available maintenance resources (classifier services, as it was shown) and explicit representation of knowledge for initial data pre-processing.

Service descriptions allow changing service set easily. Diagnostic knowledge, first, allows automated maintenance system activity and, secondly, it makes possible reuse of learnt knowledge (presented in specific classification model as data, rules, etc.) Newly installed device that has no historical data yet can use classification ontology of some other device of the same type.

We use application of Semantic Web technology for maintenance data description, service component description, representation of classification knowledge. Those descriptive data pieces have to be in some common format for the whole maintenance system. *RDF* and its derivatives are going to be a perfect basis for that.

5.3. Ontology-Based Diagnostic Services Integration

To be able to integrate heterogeneous resources and services over the Web, we have to describe them in common way based on common Ontology. In considering of resources in industrial product's maintenance domain,

we distinguish such resources as: smart devices, which are represented like services of their alarm or control systems (or some software interface); set of diagnostic services or classifiers; platforms, which are represented like clusters or collection of elements; human, which can be represented by some special service; large enterprise's systems; and etc. This ontology-based annotation must comprise not only a resource's description (parameters, inputs, outputs), but also many other necessary things, which concern their goals, intentions, interaction's aspects and etc. "Ontology-based" means that we have to create ontologies to be used for all of such resources.

Each service represents three-level resource: input parameters, "black box" (service engine), output parameters. Since all services are heterogeneous, we have a need to describe each resource (service) via common ontology and create common shell (OntoShell), which will provide such common description and will make transparent real service realization. OntoShell is a main core of such integration environment. Now as OntoShell has been elaborating in the scalable and modular manner, so it may represent mediation platform for the set of adapted, semantically annotated resources (cluster of OntoShells). OntoShell is a shell (frame) for these resources, which is a mechanism for making ontological description and providing interoperability. One of the important OntoShell's parts is an OntoAdapter for resources. If we are talking about transformation of existing resources to semantically enable ones, then we have to develop the mechanisms for accessing the resources. Since the resources are developed based on different specific standards on both content (WSDL, C/C++ (dll), Java, SQL Server, DCOM, CORBA etc.) and transport levels (TCP, HTTP, RMI, etc.). In this case we have to design and develop corresponding software modules (OntoAdapters) for semantic, content and transport levels. It will be construction blocks, which will fill OntoShell depending on resource's description.

Set of services represents a three-level automated diagnostic system. First level is represented by alarm system of device (WatchDog), which signalizes about changes of the normal device state. Main maintenance diagnostic system (service) forms second level of the system via the preliminary (initial) diagnostics. It contains preliminary diagnosis classification via the ontology of the device condition. The result of such initial classification is making decision what kind of the classifier (diagnostic service) has to make further data processing. This system initiates the third level of diagnostic for making decision about precise diagnosis. Request of the further classification is being sent to respective classifiers of the local centralized diagnostic system directly taking into account a probability of classifier's belonging to the class of the problem.

5.4. Semantic Peer-to-Peer Discovery of Maintenance Services

Within OntoSert.Net concept a peer-to-peer architecture of global network of maintenance web-services is assumed. The architecture provides support for registration of maintenance services profiles based on ontology shared within the network. The profiles are registered in local repository. The architecture includes a platform steward module, which implements semantic search engine. The engine searches among local maintenance services those, which profile corresponds to a query according to semantic match procedure. A Steward also implements peer-to-peer functionalities like query forwarding and sending, neighbour registration, etc.

When Platform Steward makes a decision about necessity of using of external Maintenance Services it sends formalized query to the neighbour platforms. Such necessity can occur if the requested Maintenance Service is absent on the local platform or cannot provide sufficient performance, or if Steward needs 'opinions' of other Classifying Services, e.g. during learning. If the neighbours can't satisfy the query, they forward it to their own neighbours and so on. Thus, the query can roam through many platforms in Global Network and the probability to find the required Service is very high.

When some platform receives query, which it can satisfy, it sends the response to the query initiator about its location. The query initiator can collect some number of such responses - a list of potential partners.

To increase the efficiency of search of Maintenance Semantic Web-services and its automated nature (initiators of the search are Smart Devices) the system should inherit the concepts of Semantic Web. That means:

1. Development of common Ontology for the Global Network which contains a classification of Maintenance Services in a hierarchical tree and explicit definition of relations between such classes. Another option might be to provide possibilities to manage several pre-existing ontologies, like in (Mena *et al*, 2000).
2. Every platform creates a local repository of profiles of available Maintenance Services based on ontology.
3. Each Platform Steward must have Semantic Search Engine, which will find a semantic match between query and each profile in the local repository.
4. Each Platform Steward must have a Semantic Query Engine, which composes formalized queries.

Thus we are using combination of centralized architecture for service discovery within the platforms of services and peer-to-peer architecture for service discovery across the platforms; see e.g. (Arumugam *et al*, 2002). A Platform Steward provides centralized capabilities when manages service discovery within its internal platform and in the same time it can behave in peer-to-peer manner when interacts with external service platforms.

6. CONCLUSIONS

The goal of this paper is to provide possible implementation framework for Web services that automatically follow up and predict the maintenance needs of field devices. Concept of a "mobile service component" supposes that any component can be executed at any platform from the Service Network, including service requestor side. This allows delivering not only a service results but also a service itself. Mobile service component carrier (agent) can move to a field device's local environment (embedded agent platform) and perform its activities locally. Service components improve their performance through online learning and communication with other components. Heterogeneous service components' discovery is based on semantic P2P search. The paper contains very basic challenges related to Web services for smart devices and partly related implementation issues. More research efforts are needed to proof some of concepts mentioned in this paper.

ACKNOWLEDGEMENTS

Author is grateful to Tekes (National Technology Agency of Finland) and cooperating companies (Agora Center, University of Jyvaskyla, TeliaSonera, TietoEnator, Metso Automation, Jyvaskyla Science Park) for the grant supporting activities of SmartResource project. I am also grateful to Dr. Jouni Pyotsia from Metso Automation for useful consultations and materials. The same concerns colleagues from Industrial Ontologies Group (O. Kononenko, A. Zharko and O. Khriyenko) for useful discussions related to the implementation issues.

REFERENCES

1. Arumugam, M., Sheth, A., & Arpinar, B. (2002). The Peer-to-Peer Semantic Web: A Distributed Environment for Sharing Semantic Knowledge on the Web. In: *Proceedings of International Workshop on Real World RDF and Semantic Web Applications*, Hawaii.
2. Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 284(5), 34–43.
3. Bussler, C., Fensel, D., & Sadeh, N. (2003). *Semantic Web Services and Their Role in Enterprise Application Integration and E-Commerce*, URL: http://www.gvsu.edu/ssb/ijec/announcements/semantic.doc
4. Chandrasekaran, B., Josephson, J., & Benjamins, R. (1999). What Are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems*, 20-26.
5. Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Payne, T., & Sycara, K. (2002). DAML-S: Web Service Description for the Semantic Web. In: *The First International Semantic Web Conference (ISWC)*.
6. Fensel, D., Bussler, C., & Maedche, A. (2002). A Conceptual Architecture of Semantic Web Enabled Web Services. *ACM Special Interest Group on Management of Data*, 31(4).
7. Gibbins, N., Harris, S., & Shadbolt, N. (2003). Agent-based Semantic Web Services. In: *Proceedings 12-th International World Wide Web Conference*.
8. Kagal, L., Finin, T., & Peng, Y. (2001). A Framework for Distributed Trust Management. In: *Proceedings of IJCAI-01 Workshop on Autonomy, Delegation and Control*.
9. Lawrence, J. (2003). Embedded FIPA Agents. In: *Agentcities*: *Agent Technology Exhibition, Barcelona*, URL: http://www.agentcities.org/EUNET/ID3/documents/exh_program.pdf.
10. Mena, E., Illarramendi, A., Kashyap, V., & Sheth, A. (2000). OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation Across Pre-existing Ontologies. *International Journal on Distributed and Parallel Databases*, 8(2), 223-271.
11. Metso (2003). Neles FieldBrowser™ System for Field Device Predictive Maintenance. Metso Automation Tech. Bulletin, URL: http://www.metsoautomation.com/.
12. Ontologies (2003), Ontologies in Agent Systems, URL: http://oas.otago.ac.nz/OAS2003.
13. Paolucci, M., Kawamura, T., Payne, T., & Sycara, K. (2002). Importing the Semantic Web in UDDI. In: *Proceedings of Web Services, E-business and Semantic Web Workshop*.
14. Pyötsiä, J., & Cederlöf, H. (1999). Advanced Diagnostic Concept Using Intelligent Field Agents. *ISA Proceedings*.
15. Satoh, I. (2001). Mobile Agent-Based Compound Documents. In: *Proceedings of the 2001 ACM Symposium on Document Engineering*, 76-84.
16. Sensodec (2003). Sensodec 6C for Paper: Metso Automation Tech. Bulletin, URL: http://www.metsoautomation.com/.
17. Terziyan, V., & Puuronen, S., (1999). Knowledge Acquisition Based on Semantic Balance of Internal and External Knowledge, *Lecture Notes in Artificial Intelligence*, 1611, 353-361.