2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Semantic Web Services in the Development of Distributed Control and Automation Systems

Kleanthis C. Thamboulidis, *Member, IEEE,* Giannis V. Koumoutsos, and George S. Doukas

Abstract— **Currently available tools for the development of the next generation distributed control and automation systems, adopt traditional architectural styles and do not cover the whole requirements of the development process. One of the major drawbacks of traditional Engineering Support Systems (ESSs), which is extensibility, can be addressed by providing a public domain framework that can be easily extended. In this paper, a service-oriented architectural framework is adopted to provide the infrastructure for the exploitation of service-oriented computing in the development process of factory automation systems. Features required in the development process are implemented as web-services and published in the public domain, so as to be used on demand by control engineers to construct their projects' specific ESSs. The infrastructure required to build a web-service based ESS is presented. Specific web-services are presented and the way that these services affect the development process is discussed. The need of a common field device model in the context of this approach is discussed and an ontology-based framework for such a device model is defined.**

I. Introduction

As the needs of industrial environments increase and become more demanding, Distributed Control Systems (DCSs) are developed using hardware and software components of more than one manufacturer. Since these different components must interoperate, the need for a common domain model for the simple and user-friendly integration of components is more than mandatory in today's complex automation systems.

The IEC 61499 standard [1] in order to decrease the productivity gap in DCSs development, has introduced the concept of the Engineering Support System (ESS), a toolset that should guide the control engineer through the development process. An ESS must allow the designer to have a global overview of the system and to be able to handle components of different manufacturers and fieldbuses. The IEC61499 also introduces the Function Block (FB) construct and the FB network diagram as the main design-phase artifacts of the proposed FB model.

Prototype implementations of ESSs are currently available [2] [3] [4] [5] and a number of projects are under way for the development of such ESSs (http://sourceforge.net/projects/oooneida-wb/, http://sourceforge.net/projects/torero). These ESSs are mainly based on a monolithic proprietary toolset and their objective is to assist the control engineer in constructing FB type and FB network diagram specifications, system layer models, validating the design specs, and deploying and executing complex DCSs.

However, these toolsets cannot fully support an effective development process. Control engineers need improved techniques, methodologies and tools to better support the analysis, design, debugging, validation, deployment and verification of the system and currently available ESSs do not fully cover these requirements [6]. Even more, control engineers will have to select the toolset that best fits their development requirements and in most of the cases the existing or under development tools do not address all of these needs. Extensibility in the form of extending these toolsets to suit project specific needs is currently slightly addressed by the available toolsets.

The control engineer to effectively address the complex development process of the next generation agile DCAs wants to pay only for the resources actually used to solve the specific problem, and monolithic environments do not cover this requirement. What is needed is an Engineering Support Environment (ESE) where the requirements of the control engineer for the development process will have the principal role [6]. Based on these requirements the control engineer should be able to set up and customize a project-specific ESE by easily integrating through plug-and-play the desirable features that should be provided through a SOA-based framework.

Descriptions of ready-to-use components such as FB types and IEC61499-compliant devices are expected to appear in the web, as soon as the IEC61499 will be adopted by industry. If this information would be constructed in the current traditional way, i.e., using presentation languages such as HTML, control engineers should use their web

Manuscript received September 15, 2006. This work has been co-funded in part from the European Union by 75% and from the Hellenic State by 25% through the Operational Programmae Competitiveness, 2000-2006, in the context of PENED 2003 03ED723 project.

K. Thramboulidis is with the Software Engineering Group, Electrical & Computer Engineering, University of Patras, 26500 Patras, Greece (corresponding author: phone: +30 2610 996436; fax: +30 2610 996820; e-mail: thrambo@ece.upatras.gr).

G. V. Koumoutsos and G. S. Doukas are with the Software Engineering Group, Electrical & Computer Engineering, University of Patras, Greece (e-mail: {koumouts,gdoukas}@ece.upatras.gr).

browsers to search for the specific FB types and field devices, required by their systems. However, this information is very difficult if not impossible to be utilized by ESSs to semi-automate the development process. Service Oriented Architecture (SOA) [7] and Semantic Web [8] provide a solution to this problem. Technologies of the semantic web, such as OWL [9], can be exploited to formalize component descriptions and make them machine-readable so that they can be more easily analyzed by ESSs to assist the control engineer in the decision making processes involved in system development. Domain models for constituent components such as FB types, field devices, etc., can be constructed in the form of ontologies, uploaded and linked into the web, so custom ESSs can link and utilize them.

In this paper, the infrastructure required to build a web-service-based ESS is presented. A service-oriented architectural framework is adopted for the exploitation of service-oriented computing in the development process of factory automation systems. Features required in the development process are implemented as web-services and published in the public domain, so as to be used on demand by control engineers to construct their projects' specific ESS. Specific kinds of web services are presented and the way that these services affect the development process is discussed. The need for a common device model in the context of this approach is discussed. An ontology-based framework for such a device model is defined and a prototype implementation to demonstrate the applicability and usefulness of the proposed approach is presented.

To our knowledge there is no other work at the moment towards the direction of utilizing SOA for the definition of an engineering environment in the form of an extended ESE that will exploit the advantages of semantic web.

The remaining of this paper is organized as follows. In the next section, the use of SOA and semantic web in factory automation is briefly discussed. In section 3, the proposed service-oriented framework for ESS is presented. Section 4, focuses on the system layer modeling, to present a concrete way of using the proposed framework; the need for a common device model is discussed and a solution to this problem based on semantic web is proposed. A prototype implementation is presented in section 5 and finally the paper is concluded in the last section.

II. Related work

Other research groups are already exploiting SOA, Web services and semantic web in factory automation [11][12][13]. The SIRENA approach [11] intends to create a service-oriented framework for specifying and developing distributed applications in diverse real-time embedded computing environments. The use of semantic Web Services as a means to address the challenge of rapid reconfiguration of manufacturing systems required in order to evolve and adapt to mass customization is proposed in [12]. Except from the fact that no specific framework is proposed, a proof of concept is also not given. A dynamic ontological definition of generic industrial resource is defined in [13], to allow flexible management, maintenance and monitoring of industrial processes.

The idea of presenting a device model as an ontology is not new. Research groups are constructing such ontologies for other domains such as mobile communications [14]. Most of these works are based on the Fipa device specification [15] and propose extensions to fully cover their domains [16]. Others have small descriptions of such a modelling attempt in their research as part of extended frameworks in visualization for collaborative planning [17], in sharing conceptual engineering knowledge [18] and in manufacturing grids [19].

III. A service-oriented framework for ESSs

The SOA framework [6] proposed as an extension of Corfu [3] and Archimedes system platform [4], is utilized and further extended to cover the system layer development process as well as the deployment and re-deployment of the application layer components to the run-time infrastructure. The proposed service-oriented framework, shown in fig. 1, intends to enable control engineers to set up and customize the ESS that best fits with the needs of their project. The control engineer instead of buying (or developing) software components and bind them together to form a custom ESS, will construct the project-specific ESS as an orchestration of web services that are only used and bound together at the time of use of the particular feature of the ESS. Using this infrastructure control engineers can also implement their own desirable features and incorporate them into their custom engineering support environment. This provides a powerful and flexible framework for customizing and extending the environment to address the control engineer's particular requirements. It enables the control engineer to construct an ESS by using services by multiple suppliers to meet the needs of the specific project.

Developers of web-services will exploit the device model to semi-automate the development process of DCSs regarding: a) the design process of the system layer, b) the deployment process, and c) the verification process.

During the system layer design phase, the control engineer searches (1 in fig. 1) through the ESS the web (device vendor's repositories) to locate devices that meet required QoSs. These QoSs are imposed either by controlled process (e.g. number and type of available process parameters to be sensed or actuated), or by the components of the control application (e.g. number and functionality of FB types used). Through semantically annotated web-services, the control engineer performs an ontological search based on concepts with which he is familiar, since these are described in the specific domain device ontology (2). He may have access to basic characteristics of the device as well, as that is also included in the specific device model

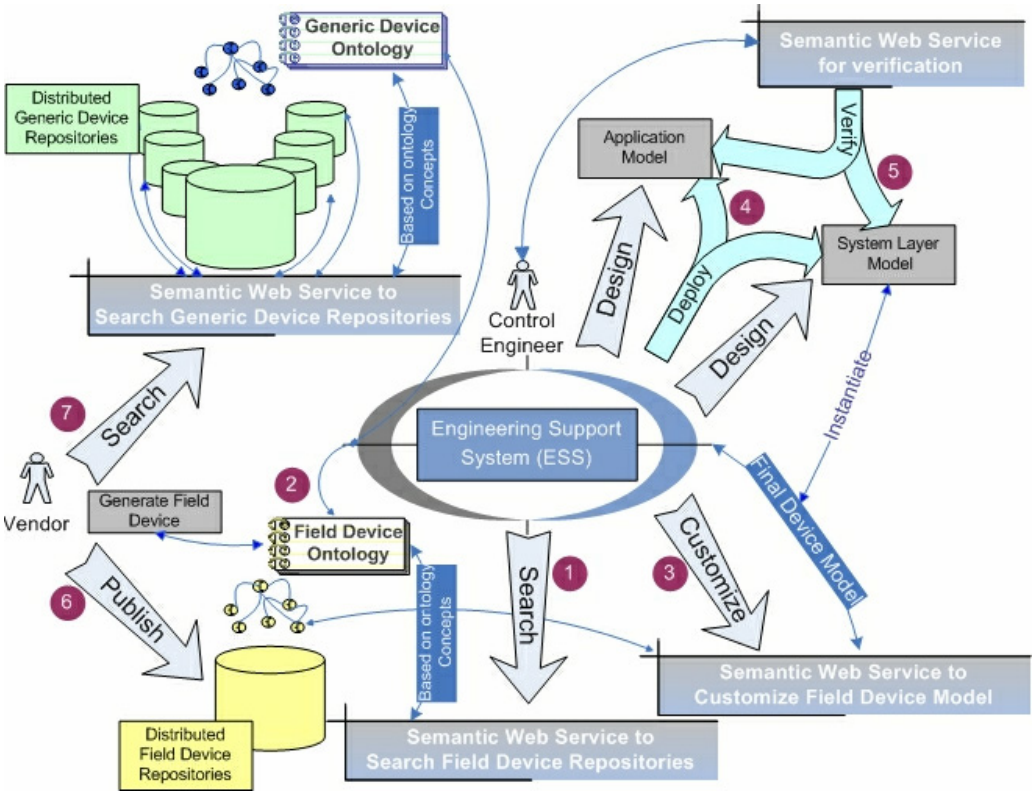

Fig. 1. Conceptual model of the proposed semantic web based framework.

constructed by the vendor. This is one of the advantages of introducing levels of abstraction in the design of the device model.

Devices are usually described in terms of optional configurations. A device for example may be configured to have various I/Os (analog/digital) or support various operating systems. A specific web service that will have the ability of manipulating ontologies relieving the control engineer from this burden, will allow him to describe the desired configuration (3) that will be used for the specific application. Choices will be made in a user friendly way and the web service will create the device model of the defined configuration. The so created device model can be downloaded and used for the design of the system layer diagram.

The device model specification plays also a significant role during the deployment process (4). That's when decisions must be made about the distribution of the application's components and the generation of the application's implementation model. During this process, the device model can be automatically updated, with the use of rules and rule-engines, every time its available resources change, as for example when new components are downloaded. Based on this the control engineer will always be aware of the remaining recourses. Specific functionality provided by the ESS may be utilized to search for possible alternatives that satisfy the required by the application layer components QoSs.

Finally, the device model may be utilized through the verification process (5) of the design model. Device descriptions in the form of knowledge bases for the specific project will be stored in the project's repository and will be exploited by design-model analysis and verification tools to verify that the application's design diagrams, as well as the planned deployment scenarios are implementable. Later on, and after the verification of the design models of the DCS the real world devices can be bought through a web service.

IV. System Layer Modeling

The system layer is considered as an aggregation of interconnected field device instances on which the components of the control application will be assigned. Interconnecting edges provide the infrastructure required for the realization of component interactions that cross device boundaries. In the context of this work we examine the model of the field device and the way that this model can semi-automate the development process.

A. The need for a common Device Model

A large number of field devices of different vendors are used for the control and automation of manufacturing systems. These devices are usually proprietary and device vendors provide specific proprietary tools to handle them. This is why many different tools must be handled in the life cycle phases of DCAs [20]. Information should be exchanged between these tools making the task of

integration very difficult. The large number of different device types and suppliers within a given control system makes the configuration task difficult and time consuming.

It is also clear that the different proprietary device tools coming from a variety of device vendors cannot be consistently integrated into an ESS. The problem of configuring and parameterization of different field devices during the operation diagnosis, parameter tuning, processing purposes, etc. constitutes one of the most important factors for not moving to an open market in factory automation. This problem was recognized and different device models were constructed to address this demand [21][22][23][24]. Device Description Languages (DDLs), already support the specification of field devices, with HART DDL[21], Profibus device description [23], and Foundation fieldbus DDL [22] being among the most important. These notations are used to represent the properties of a field device in a proprietary machine-readable format to be used by proprietary engineering tools during the development phase of the DCAs. The specification is also used during the DCSs operation phase.

However there is no common model for the device specification and the above notations result in incompatible device specifications. A device model consistent with current software engineering practices should be defined to enable the new generation ESSs to automate the development and deployment process to a large extent. Operations to be supported by the field device model include the following:

- Select the appropriate device that meets the required by the software components of the control application QoS characteristics.
- Configure the device to meet the requirements of the current implementation.
- Semi-automate the deployment and re-deployment process.
- Create the dynamic model of the device that represents the device at run-time.

B. A UML based Device model

The Unified Modelling Language (UML), the defacto standard for modelling software intensive systems, was used to create a reference model for the device specification. Part of this model is shown in Fig. 2. The UML profile for Schedulability, Performance, and Time Specification [25] was utilized for the modelling of resources that constitute a device. According to this the resource is considered as the common basis for modelling all quantitative aspects of software systems. A resource is modelled as a server that provides one or more services to its clients [26]. The physical limitations of a resource are represented through its QoS attributes. The QoS concept is used in the context of this framework to establish a uniform basis for attaching quantifiable information to UML models. QoS information represents directly or indirectly the physical properties not only of the application's components (required QoS) but also those of the hardware and software infrastructure used to execute the control application (offered QoS). Clients competing for the same resource greatly complicate the analysis of the model.

UML's extensibility mechanisms can be used to create a more expressive model for the device. The construct of stereotype is used to define a specialization of the class construct to add the semantics of the device to the class generic UML construct. Additional constraints and tagged values are used to represent additional attributes of the device.

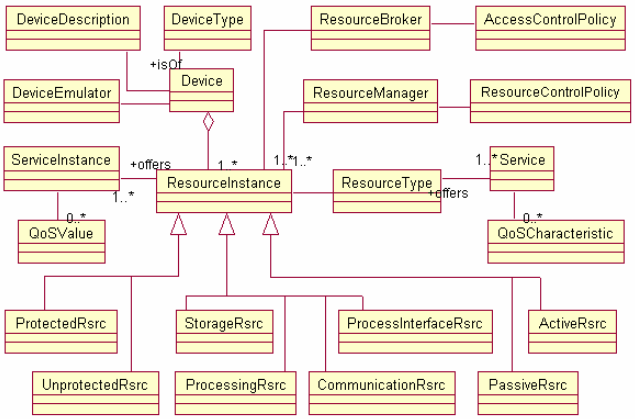

Fig. 2. Part of the proposed device model expressed in UML notation.

C. Towards a Device Ontology

Semantic Web technologies can be exploited to increase the reusability of the so created domain models and make these models available through the web. This allows device vendors to locate a suitable device model on the web and simply reuse or extends it, instead of developing their own model. By reusing these models, different web services can share results and data much more easily and simplify their integration to form a consistent ESS. Semantic web is used as a platform on which the domain specific device model will be created in such a way that sharing and reusing by many different applications across the web will be the primary objective.

The device ontology represents the common conceptualization that is required to increase the degree of automation in the system layer development process. This device ontology should define, in a machine processable format, the meanings of concepts concerning storage, processing and communication capabilities of the device and should facilitate the processing of information of heterogeneous devices in the design phase of the system layer diagram.

A prototype device ontology was constructed to demonstrate the effectiveness of this approach. Since, collecting and categorizing all concepts in an all-in-one ontology is a very difficult if not impracticable issue, the focus was to construct a basic ontology upon which extensions and mappings can be later defined. The FIPA device ontology [15] was utilized for the definition of the

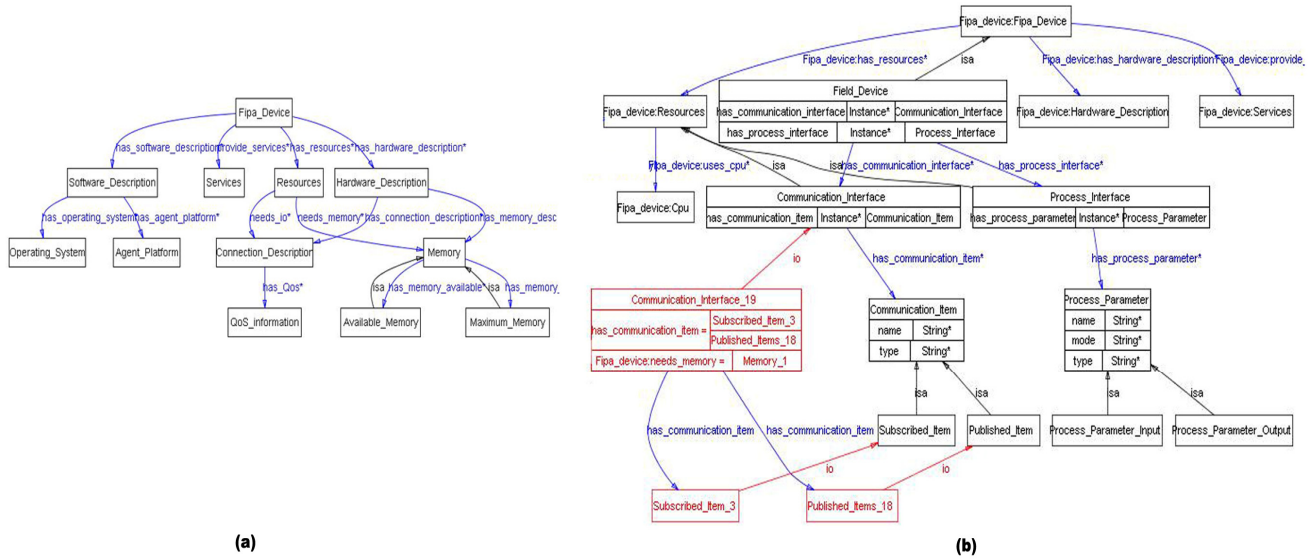

Fig. 3. (a) Part of the FIPA-based Device ontology, (b) Part of the proposed field device ontology created by protégé.

core of our device ontology. Figure 3(a) shows part of the ontology developed with ontoviz tool of protégé 3.2, to capture the general concepts of a device. Fipa device classes such as Software and Hardware_Description are shown, as well as other important concepts we have added such as Resources which are properly linked to Memory and Connection_Description classes with needs_io and needs_memory object properties respectively.

The so constructed ontology was next extended with new concepts and properties captured in the device model shown in figure 3, to get a more specific description of the IEC-compliant field device. All mappings between extended and basic FIPA concepts are also done in order to justify our choice on using this starting level of abstraction. This will also help us to refer to a field device with its basic properties as a simple device and reason over them as it will be described below. Our ontology covers the most important and general concepts of a field device, leaving room for extensions where needed and translations and mappings with the use of advanced semantic artifacts like rules and translation ontologies.

Figure 3(b) presents part of the field device ontology as displayed by protégé. This ontology incorporates concepts from the imported Fipa_Device ontology and adds new concepts required to better describe a field device. An example of the integration with the Fipa_Device is shown form the fact that Field_Device is a kind of Fipa_Device and, Communication_Interface and Process_Interface, which are kinds of Fipa:Resources. This integration provides flexibility to our ontology since it allows the creation of mappings between instances of concepts belonging to different abstraction levels.

A device ontology can be utilized in multiple ways, for different purposes taking advantage of the multiple levels of abstraction captured by the ontology, as shown in fig. 3(b).

Device vendors will populate the ontology with data to create their device description that will be stored either in their local meta-device repository or to third party remote meta-device repositories (6), as shown in Fig. 1.

Specific domain device-vendors may utilize, during the development process of their devices, the lower layers of abstraction, i.e. those expressed by the FIPA device ontology. Through semantically annotated web services they can search the web (7) to locate devices that cover their needs in terms of computational power, storage capacity and communication capabilities.

The selected device will be used to implement the specific domain device, such as an IEC61499-compliant device, that will provide advanced services for the specific application domain. For example, an IEC61499-compliant device will provide services for downloading and instantiating FB types, creating event and data connections, implementing inter-device connections and so on. The result of the development process will be a real world device accompanied by a new device description that will be based on the initial device description and the population of the device ontology of the specific application domain in which the new device belongs, as shown in Fig. 1. This specific domain device ontology has to be created and standardized to allow interoperability in software tools and generated descriptions.

V. A PROTOTYPE IMPLEMENTATION

To demonstrate the applicability of the above described processes and technologies in factory automation a prototype implementation was developed. A web service through which FB types can be located and downloaded for free and a service through which an XML FBType description can be translated into a C++ executable model, were implemented. The first of them is a key service for increasing reusability. It allows the control engineer to locate already available FB type specifications and use them in the development process of its control applications. It also

allows vendors to develop generic and specific FB types and advertise them for sale through the web infrastructure, as sown in Fig.1. The second web service covers the need of transforming the FB type design specification to an executable specification (implementation model) for the specific platform.

Both, web-services and corresponding prototype clients were implemented in Java 1.5.0. Eclipse 3.1.2 with Web Standard Tools plugin was used to construct, following a bottom-up approach, a web service selecting the methods from our implementation that would be exposed publicly. Version 1.0.2 of the plugin was utilized for the development of the services, while version 1.0.1 was used for the development of prototype clients. An interface and a WSDL document to describe the service were generated. A client ready to invoke the service, if provided with the appropriate parameters, was constructed using this WSDL description. The web-services are currently deployed on Tomcat 5.5.17 servlet container, while the API used to handle the SOAP messages is Apache Axis.

These web services have been published in a private UDDI to allow for any user to locate and use these services through a WSDL interface which is also published at the same UDDI.

Apache juddi, which gives an interaction interface through filling XML documents with the appropriate information, was utilized to implement our UDDI service. The provided UDDI with all the prototype implementations can be reached at: http://seg.ece.upatras.gr/seg/dev /SOA4DCS.htm

VI. CONCLUSIONS

Currently available or under development IEC-compliant Engineering support systems do not provide the flexibility imposed by the development process of complex tomorrow's agile factory automation systems. The most important limitations to this inability are introduced by the traditional architectural paradigms that were utilized to construct them.

The service-oriented architectural paradigm was adopted to define a framework for the easy integration of desirable features and their customization to form project specific ESSs. Specific web services were developed to demonstrate the applicability of this approach. For the better integration of the different web services the need of a common domain model was identified. UML was used to define such a domain model for the field device. However, to get the best in terms of interoperability and reusability from the so defined models, semantic web technology should be exploited. The prototype device ontology that was developed using protégé demonstrates the usefulness of this approach in the different phases of the development process. The described framework benefits both from the adopted service-oriented architectural paradigm and the semantic web technology to provide a promising platform for the next generation ESSs for the factory automation domain.

REFERENCES

[1] International Electro-technical Commission, (IEC), Function Blocks, Part 1 - Part 4, IEC Jan. 2005. (http://www.iec.ch/)

[2] Western Reserve Controls, Inc., W2 Series IEC61499 Development Kit http://www.wrcakron.com/IEC61499.html

[3] CORFU ESS, Available on-line, http://seg.ece.upatras.gr/corfu

[4] Archimedes System Platform, Available on-line, http://seg.ece. upatras.gr/MIM

[5] ICS Triplex ISaGRAF, Commercially Available IEC 61499 Software, http://www.icstriplex.com/

[6] Thramboulidis,K., Koumoutsos,G., Doukas,G., "Towards a Service-Oriented IEC 61499 compliant Engineering Support Environment", 11th IEEE Int. Conf. on Emerging Technologies and Factory Automation, (ETFA'06) Sept 2006, Prague

[7] Thomas Erl, Service-Oriented Architecture, Concepts, Technology and Design, Prentice Hall, 2005.

[8] W3C, Semantic web, http://www.w3.org/2001/sw/

[9] W3C,OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features/

[10] Bichler, M., Kwei-Jay Lin; "Service-Oriented Computing", IEEE Computer Volume 39, Issue 3, March 2006 Page(s):99 – 101

[11] F. Jammes, H. Smit, "Service-Oriented Paradigms in Industrial Automation", IEEE Transactions on Industrial Informatics, II, vol. 1, no. 1, Feb 2005.

[12] Lastra, J.L. Delamer, M. Semantic web services in factory automation: fundamental insights and research roadmap, IEEE Transactions on Industrial Informatics, Vol. 2, Issue 1, pp. 1- 11, Feb. 2006

[13] Kaykova O., Khriyenko O., Naumenko A., Terziyan V., Zharko A., "RSCDF: A Dynamic and Context Sensitive Metadata Description Framework for Industrial Resources", In: Eastern-European Journal of Enterprise Technologies, Vol.3, No.3, June 2005.

[14] H. Chen, T. Finin and A. Joshi, "Semantic Web in the Context Broker Architecture", IEEE Conference on Pervasive Computing and Communications (PerCom), Orlando, March, 2004.

[15] FIPA Device Ontology Specification, http://www.fipa.org/specs /fipa00091/XC00091C.pdf

[16] A.Bandara, T.Payne and et.al, "An Ontological Framework for Semantic Description of Devices", The 3rd International Semantic Web Conference, Japan, November 2004.

[17] Queiroz N., Lino and Austin Tate, "A Visualisation Approach for Collaborative Planning Systems Based on Ontologies", Proceedings of the 8th International Conference on Information Visualization 2004.

[18] Mizoguchi, R. Kitamura, Y., "Foundation of knowledge systematize-tion: Role of ontological engineering", Industrial Knowledge Management - A Micro Level Approach, Springer-Verlag, 2000.

[19] Shi Shengyou, Yang Haicheng, Mo Rong, Chang Zhiyong, Chen Zefeng, "Research on modeling resources based on Web service technologies in manufacturing grid", IEEE International Conference on e-Business Engineering, 2005. ICEBE 2005.

[20] Simon, R., Peter Neumann, Diedrich, Ch. Riedl, M., "Field Devices - Models and their Realizations", IEEE international Conference on Industrial Technology, Bangkok, Thailand 2002.

[21] "The HART book 9", http://www.thehartbook.com/articles/ h9ddl.asp.

[22] FF-94-890 PS 1.0 Preliminary Standard, Fieldbus Foundation 1995.

[23] The International Open Fieldbus Standard EN50170, "Profibus Guideline – Specification", PNO Draft 1998.

[24] Thramboulidis, K., Prayati, A., "Field Device Specification for the Development of Function Block Oriented Engineering Support Systems", IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA 01), French Riviera 2001.

[25] OMG, "UML Profile for Schedulability, Performance, and Time Specification, Ver. 1.0, September 2003.

[26] Selic, Bran, "A Generic Framework for modeling resources with UML", IEEE Computer, June 2000.