Applications of Dynamic Deployment of Services in Industrial Automation

Gonçalo Candido[1], José Barata[1], François Jammes[2], and Armando W. Colombo[3]

[1] UNINOVA – Universidade Nova de Lisboa, Portugal
{gmc,jab}@uninova.pt
[2] Schneider Electric – Corporate R&D, 38TEC – Grenoble, France
francois2.jammes@schneider-electric.com
[3] Schneider Electric Automation GmbH, BU Automation, SysCo, Germany
armando.Colombo@schneider-electric.com

Abstract. Service-oriented Architecture (SOA) is becoming a de facto paradigm for business and enterprise integration. SOA is expanding into several domains of application envisioning a unified solution suitable across all different layers of an enterprise infrastructure. The application of SOA based on open web standards can significantly enhance the interoperability and openness of those devices. By embedding a dynamical deployment service even into small field de- vices, it would be either possible to allow machine builders to place built- in services and still allow the integrator to deploy on-the-run the services that best fit his current application. This approach allows the developer to keep his own preferred development language, but still deliver a SOA- compliant application. A dynamic deployment service is envisaged as a fundamental framework to support more complex applications, reducing deployment delays, while increasing overall system agility. As use-case scenario, a dynamic deployment service was implemented over DPWS and WS-Management specifications allowing designing and programming an automation application using IEC61131 languages, and deploying these components as web services into devices.

Keywords: Service-oriented Architecture, Device Model, Services Dynamic Deployment, DPWS, WS-Management.

1 Introduction

Service-oriented Architecture (SOA) is a major focus of interest from device level to high level IT [1] [2] [3] [4]. SOA promises to lead to near-perfect applications in which every function is implemented and exposed as a service possible to be discovered and used by other network element. SOA establishes an architectural model that aims to enhance the efficiency, agility, and productivity of an enterprise by positioning services as the primary means through which solution logic is represented in support of the realization of strategic goals associated with service-oriented computing. The continuous convergence between computing and networking areas, enabled by the advances in semiconductor and transmission technology, allows new approaches to communication between systems and devices, in particular, embedded devices. The expansion of SOA

L.M. Camarinha-Matos, P. Pereira, and L. Ribeiro (Eds.): DoCEIS 2010, IFIP AICT 314, pp. 151–158, 2010.

approaches into different domains of application promises to deliver a complete cross-level and cross-domain connectivity over the same technology paradigm.

At crescent rhythm, Internet technology is emerging as the basic carrier for interconnecting electronic devices in widely diverse domains of application [5] [6] [7]. One of the most promising approaches concerns the application at device level where the usage of high level service-based communications infrastructure allows completely innovative advances. Some research opportunities were already mapped in [8].

The goal of this document is to summarize a SOA-based solution for the industrial automation domain evidencing the role of the services dynamic deployment feature as a fundamental tool to enhance systems agility at device level. The concept of agility in this context implies being more than flexible or lean. Flexibility refers to a company that can easily adapt itself to produce a mostly predetermined range of products, while lean essentially means producing without waste. On the other hand, agility corresponds to operating efficiently in a competitive environment dominated by change and uncertainty [9]. Agility is a fundamental requirement for modern production companies in order to face challenges provoked by the globalization, environmental and working conditions regulations, improved standards for quality and fast technological mutation [10]. However, global company agility is always limited by its least agile building block – all levels of the CIM pyramid, from ERP to field device level, need to be agile and interact in a seamless and synchronized manner.

At organization level, managers already noticed that they need to cooperate with other organizations in order to remain competitive [11]. Although several work has been done on the topics of virtual organizations agility of production and/or collaboration to deal with unexpected demands and volatile markets at level [12] [13], this agility can only be achieved if all organization levels are also equally agile. The automation devices level plays a fundamental role, since a device is the last frontier where high level process workflows are transformed into a structured collection of physical actions to be performed in a particular sequence – device control and management aspects are here crucial to support above levels agility.

The next section clarifies the importance of this work as an innovative contribution to future exploitations in the domain SOA-ready devices in industrial automation; section 3 exposes a device model comprising the dynamic deployment feature; section 4 pictures the overall architecture comprising the previous device model features; section 5 summarizes a dynamic deployment prototype implementation for the industrial automation domain; and, at last, the final conclusions and future perspectives are presented.

2 Contribution to Technological Innovation

The approach exposed in this document represents a new trend input concerning SOA-ready devices. Specifically, in the industrial automation domain it is fundamental to understand the concrete requirements and progress expectations – how can SOA paradigm provide its own contribute here?

In the industrial automation domain, and more concretely in manufacturing systems, the major goal is to produce fast, cheap, and quality products in line with current and expected client demands while remaining agile enough to handle constant

output fluctuations, in terms of form and quantity. The interoperability and easy access to a device plays a major role when trying to accomplish previous goal.

Since the majority of high level business IT is already based over SOA technologies, it is important to research the field device domain and make it effortlessly compliant with the above infrastructure, while providing the expected functionality and performance to domain experts. Although the device model present in this document is most focused over the industrial domain, it remains abstract enough to fit other domains of application, as well as the envisaged applications of this device model in an involving infrastructure. By deploying an SOA-based middleware that already provides, besides others, discovery, identification, services invocation and eventing functionalities over an open standard, any developments over it will only need to focus on application behaviour since compliance and openness are already implicitly embedded. So, by allowing to manage and to deploy services into a device in a generic way, the system agility is increased while remaining compliant with open standards which also increase device interoperability across different vendors.

In summary, this work is expected to increase overall system agility by providing a new device model enriched with a dynamic deployment of services feature laying over open web standards possible to be extended and mapped to new domains of application.

3 Device Model

3.1 Overview

In this context, a device is to be seen as the main logical entity that abstracts an application element, while its services represent the functionalities that a particular element allows others to exploit to accomplish its own goals. A service will encapsulate a function or behavior that can be discovered and employed by any other network element in order to execute a particular task. However, it's possible to define a logical device that can be defined whenever there is a need to abstract a particular system component not explicitly associated to a physical entity. The definition remains abstract enough to be mapped to a wider range of application domains. Also, the application can be composed of several devices that interact between them through the services hosted on those with no imposed control architecture envisaged a priori.

3.2 Built-in vs. Deployed Services

The device model in Fig. 1 represents the device itself as a real-world physical entity. This physical device will already embed some built-in services that allow deploying applications but also other added-value services that allow the integrator to setup, monitor, and diagnose that particular device. These services are deployed by the device builder, being immediately available when taking a new device out from the box. They cannot be removed or modified by the end-user – if the end-user wants to add its own services, he can do it through dynamic deployment. This collection of services is considered generic to every particular range of devices. The dynamic deployment service is itself a built-in service, which allows the systems integrator to deploy its own resources i.e. logical devices and its services.

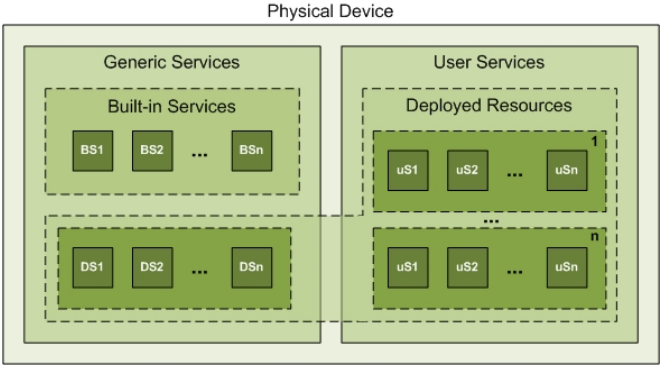

Fig. 1. Device Model

The user application will be dynamically deployed in a form of logical devices. These logical devices will represent the logical entities that can be observed from the current application, exposing their hosted services as their capabilities that others can make use of. An application can even compose several of these logical devices into several layers of increasing abstraction, in an orchestration or choreography manner – application construction based on existing building blocks. Still, it would be possible to deploy services that will enhance the functionality already provided by the current built-in services – these services are also considered generic to that particular range of devices, although they are not considered mandatory to the majority of applications. This approach allows a higher level of device customization. Also, some application functionalities can be considered generic enough and be reused across several applications, being deployed whenever they are required. For example, it would be possible to deploy into a PLC device some services that can control some common system components. This approach will avoid the need to recode those components every time they are needed and still remain consistent with previous implementations or development guideline.

4 Infrastructure

The device model presented in previous section will then fit in a bigger infrastructure that will make use of its embedded capabilities. These features at device level will subsequently enhance overall system ability in terms of interoperability and agility by laying over open web standards and allowing a fast deployment of services whenever system requirements changes oblige to. Since implementation details are encapsulated by the service itself under its interface, it will be possible to retrieve and connect to a device without knowing its current IP address, but simply searching for the required functionality - transparent interoperability. The integrator has also the ability to manage the complete lifecycle of these resources, as described in [14].

Deployed services can either be developed by the integrator or retrieved under an online repository of services - offered by device manufacturer or any other partner (see Fig. 2). When built-in services are not enough to current system requirements,

there is a need to deploy extra services that might allow a more complex control over the device, or implement a common function to a particular domain of application. This infrastructure opens the door to new business models, where a disparate range of developers sell their own services to fill common needs. Although users can develop and deploy their own devices and services resources, it would be also possible to build an application only by combining resources developed by partners and/or reused from previous projects. Besides, the clear separation between hardware and software layers will allow the same service to be deployed across dissimilar physical devices. Even if implementation changes due to variations on device firmware capabilities, the service interface remains the same for the same concrete function – a service can be substituted by an equivalent one without changing anything only by keeping the service interface.

Whenever there is a need to connect to a particular device for configuration, maintenance, monitoring or diagnosis purposes, it can be used a device with any web standards-compliant tools, such as a PDA or a tablet PC with wireless access. This allows the user motion along the platform to better check equipment status and behavior.

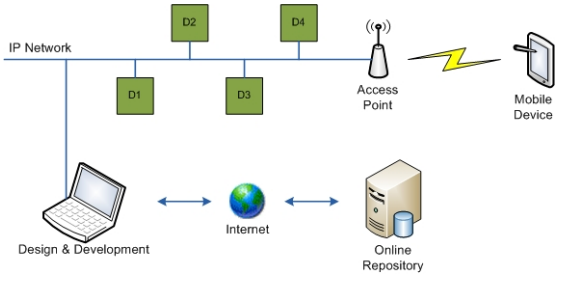

Fig. 2. Network topology example

The device generic built-in services will provide basic discovery, management and connection operations even on a new device. In the same way, the actual application will be easily discoverable and directly interoperable in the network since it is deployed in the form of several logical devices deployed across different physical devices. From the end-user point of view, the application is totally independent from the actual hardware platform – logical view of the system.

5 Industrial Automation Scenario

In the industrial automation domain, systems integrators are used to their own processes and programming languages. IEC61131 programming languages are the most used within this domain, due to its simplicity and run-time performance. The challenge is then how to turn this reality compatible with a complete SOA environment, while increasing system capacity in terms of agility, interoperability and robustness.

Devices Profile for Web Services (DPWS) is a common web services middleware and profile for devices [15], being currently ongoing a standardization process by OASIS. WS-Management [16] specification describes a general WS* based protocol

for managing systems such as PCs, servers, devices, Web Services, applications, and other manageable entities. The implementation summarized in this chapter, firstly shown in SODA project industrial demonstrator [17], will be further exploited in SOCRADES project industrial pilot application. It joints DPWS and WS-Management specifications in a single application that supports the dynamic deployment of services, besides other features already described in [14].

The automation control application is developed by abstracting different system components and coding its behavior using IEC61131 programming languages. The original component IEC61131 code is translated to PLCOpen format, being embedded in a service template that will then be deployed into the device (see Fig. 3). It should be noted that this template is independent from particular implementation details, which will be comprised in a parameter value that can be run or interpreted by device firmware. Once the component is made available and active, it will be possible to retrieve it and its services invoked or events subscribed in the network as any other DPWS device – using DPWS, the discovery and identification features are already supported in a distributed manner.

As example, operating systems "Windows Vista" and "Windows 7" are already DPWS-compliant by default, being possible to use the available device explorer to discover and interact effortlessly with any DPWS device available in the network. It is then possible to retrieve its individual metadata values, comprising not only some catalog parameters but also available services contract and a link to the device web server where the user is able to define a collection of setup, monitoring and diagnostic parameters through any traditional web browser. This simple feature already represents a major input to enhance out-of-the-box device connectivity and setup – one of the biggest customer demands.

Even though this implementation focused over industrial automation domain, it remains abstract enough to adapt to other domains of application since the implementation details are isolated from the process to deploy new services into devices.

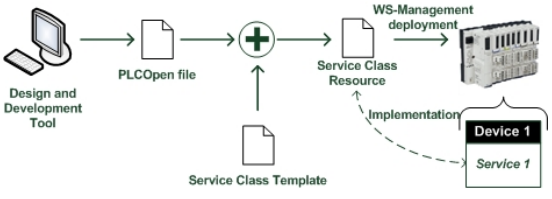

Fig. 3. Dynamic deployment process

6 Conclusions

SOA paradigm is envisaged to expand and become an essential aspect at device level, including in industrial automation domain. It is then important to create models and methodologies to extract the major inputs from this paradigm to face domain requirements.

The existence of a dynamic deployment service embedded on a device can al- low more complex applications plus the creation of new business opportunities. It

increases device interoperability and overall system agility whenever there is a need to update, manage or connect to it due to behavior adjustments or occasional downtime. Implementation intricacies are hidden under services interfaces that remain compliant with the overall SOA infrastructure, allowing an end-to-end interoperability across the different levels of the global application. These devices and services can be retrieved and used in a distributed manner across a common IP network in a transparent way.

Still, there is an important need to create new tools and methodologies to provide an easier development under this new paradigm compliant with the major open web standards. Also, training and clarification activities are also of major importance when pushing a new paradigm into a domain known to be particularly restrict as the industrial automation domain. Security and semantic aspects, as possible adaptation to other domains of application can be envisaged as future research topics over this same subject.

Acknowledgments. The work present here was developed within the scope of the European IST FP6 project "Service-Oriented Cross-layer infRAstructure for Distributed smart Embedded deviceS" (see http://www.socrades.eu).

References

1. Bloomberg, J., Schmelzer, R.: Service orient or be doomed. Wiley, Chichester (2006)
2. Erl, T.: Service-oriented architecture: concepts, technology, and design. Prentice Hall PTR, Upper Saddle River (2005)
3. Rosen, M., Lublinsky, B., Smith, K., Balcer, M.: Applied SOA: Service-Oriented Architecture and Design Strategies. Wiley India Pvt. Ltd., Chichester (2008)
4. Bell, M.: Service-oriented modeling: service analysis, design, and architecture. Wiley, Chichester (2008)
5. Jammes, F., Smit, H.: Service-oriented paradigms in industrial automation. IEEE Transactions on Industrial Informatics 1(1), 62–70 (2005)
6. Karnouskos, S., Baecker, O., De Souza, L., Spiess, P.: Integration of SOA-ready networked embedded devices in enterprise systems via a cross-layered web service infrastructure. In: 12th IEEE Conference on Emerging Technologies and Factory Automation, pp. 293–300 (2007)
7. Barata, J., Ribeiro, L., Colombo, A.: Diagnosis using Service Oriented Architectures (SOA). In: 2007 5th IEEE International Conference on Industrial Informatics, vol. 2 (2007)
8. Candido, G., Barata, J., Colombo, A., Jammes, F.: SOA in reconfigurable supply chains: A research roadmap. Engineering Applications of Artificial Intelligence (2009)
9. Goldman, S., Nagel, R., Preiss, K.: Agile competitors and virtual organizations: strategies for enriching the customer. Van Nostrand Reinhold Company (1995)
10. Lin, C., Chiu, H., Chu, P.: Agility index in the supply chain. International Journal of Production Economics 100(2), 285–299 (2006)
11. Vernadat, F.: Interoperable enterprise systems: principles, concepts, and methods. Annual Reviews in Control 31(1), 137–145 (2007)

12. Camarinha-Matos, L., Afsarmanesh, H.: A modeling framework for collaborative networked organizations. In: Network-centric collaboration and supporting frame- works. IFIP working conference on virtual enterprises, pp. 3–14. Springer, Heidelberg (2006)
13. Camarinha-Matos, L.: ECOLEAD-Achievements in Collaborative Networked Organizations. In: IFIP Working Conference on Virtual Enterprises, Guimarães, Portugal (2007)
14. Cândido, G., Jammes, F., Barata, J., Colombo, A.: Generic Management Services for DPWS-enabled devices. In: Proceedings of IECON 2009 Annual Conference of the IEEE Industrial Electronics Society (2009)
15. OASIS: Devices profile for web services version 1.1 specification (2009), http://www.oasis-open.org/committees/ws-dd
16. DMTF: Web services for management (ws-management) specification (2008), http://www.dmtf.org/standards/wsman/
17. SODA: Soda project industrial demonstrator (2008), http://www.soda-itea.org/Demonstrators/Industrial/default.html