Integration of Legacy Devices in the Future SOA-based Factory

Stamatis Karnouskos * **Thomas Bangemann** **
Christian Diedrich ***

* *SAP Research, Germany (e-mail: stamatis.karnouskos@sap.com)*
** *ifak - Institut für Automation und Kommunikation e.V. Magdeburg, Germany (e-mail: thomas.bangemann@ifak.eu)*
*** *Otto-von-Guericke Universität Magdeburg, Germany (e-mail: christian.diedrich@ovgu.de)*

AbstractFuture shop-floors are going to evolve as they need to be able to fully respond to dynamic adaptations and sophisticated interactions with the enterprise systems. This trend is fully backed up by the ever increasing capabilities of the new generation of devices that feature advanced communication as well as computational capabilities. However as the transition to the future infrastructure will be done progressively we need to make sure that the legacy devices can still be included and offer their benefits for non-interruptible business operation.

Keywords: SOA, web services on devices, DPWS

1. INTRODUCTION

Future shop-floors are expected to be service-oriented (Colombo and Karnouskos (2009)), where devices will offer their functionality as a service and collaborate. Especially if we consider the vision of Internet of Things (Fleisch and Mattern (2005)), millions of devices will be interconnected with eachother and also with enterprise systems. The increasing computing capabilities of embedded devices will allow the implementation of new software for completely novel processes. Enabled by software, the Internet of Things provides for virtually infinite integration of sensors, actuators, microsystems, mechatronic systems, and robots. The world market for technologies, products, and applications alone that are related to the Internet of Things will increase significantly from €1.35 billion to more than €7.76 billion in 2012, with average annual growths rates of almost 50% (sap (2008)).

The emerging approach is to create system intelligence by a large population of small and smart networked embedded devices at a high level of granularity, as opposed to the traditional approach of focusing intelligence on a few large and monolithic applications. This increased granularity of intelligence distributed among loosely coupled intelligent physical objects facilitates the adaptability and reconfigurability of the system, allowing it to meet business demands not foreseen at the time of design and providing real business benefits (Spiess and Karnouskos (2007)). It is expected that in the future business applications will heavily interact with the "real-world" via the optimal timely exploitation of the information offered by devices.

The use of the service-oriented architecture (SOA) paradigm, implemented through Web Service (WS) technologies, at the ad hoc device network level enables the adoption of a unifying technology for all levels of the enterprise, from sensors and actuators to enterprise business processes. The

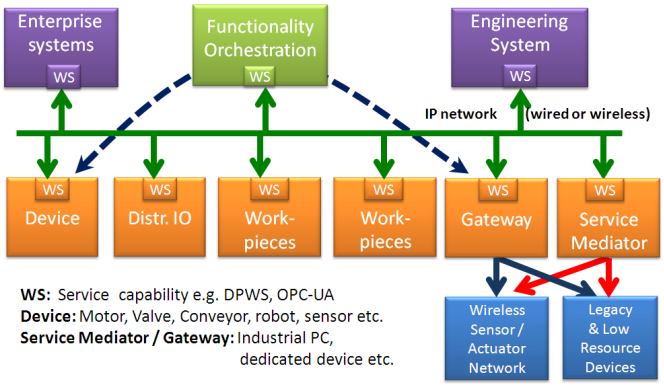

Figure 1. SOCRADES Vision: SOA based integration of the future factory shop-floor

benefits of service-orientation (soa (2008)) are conveyed all the way to the device level, facilitating the discovery and composition of applications by re-configuration rather than re-programming. Dynamic self-configuration of smart embedded devices using loosely-coupled services provides significant advantages for highly dynamic and ad hoc distributed applications. A fundamental goal is to enable the easy integration of device-level services with enterprise systems overcoming the heterogeneity and specific implementation of hardware and software of the device. This goal will require the definition of new integration concepts taking into account the emerging requirements of business applications and the explosion of available information from the device level. Of particular interest is the availability of near real-time event information, which will be used to specify new enterprise integration approaches for applications such as business activity monitoring, overall equipment effectiveness optimization, maintenance optimization, etc.

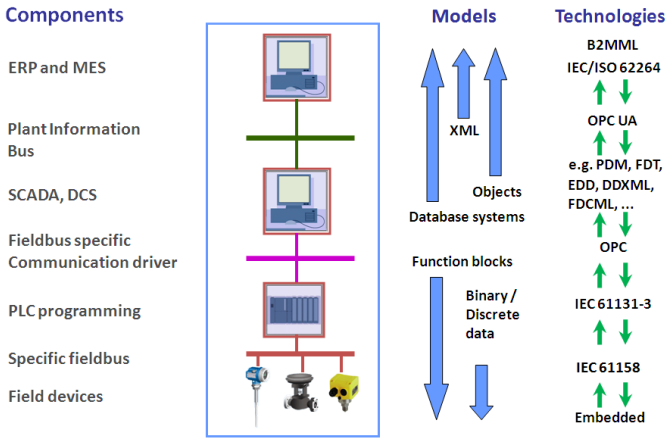

Figure 2. Example of heterogeneity in current factories

According to the vision of the SOCRADES project (`www.socrades.eu`) depicted in Fig. 1, the communication in the future shop-floor will be mainly based on wired/wireless Internet Technologies e.g. IPv6 or its variants 6lowpan (Ma and Luo (2008)) for communication, possibly (web) services e.g. DPWS (Chan et al. (2005)), OPC-UA (Hannelius et al. (2008)), REST (Liu and Connelly (2008)) at higher levels etc at least with respect to the high-level management and enterprise integration. Entities (devices, applications, etc) will communicate in a cross-layer way but heavily collaborate in mash-ups (Colombo and Karnouskos (2009)). Aggregated device-level services will interact with higher-level business processes situated at the level of business applications - in particular Enterprise Resource Planning (ERP) systems - in order to demonstrate seamless integration of device level functionality into higher-order business application scenarios.

2. DEVICE HETEROGENEITY

Current infrastructures are highly heterogeneous (Fig. 2) and a tremendous amount of effort has been invested in tackling it. However still we do not have long-lasting future-compatible solutions. The future factory shopf-floor will also be heterogeneous in devices, protocols etc, but this will be optimally hidden e.g. via SOA, and will not pose a significant integration problem any more.

Device integration based on SOA has already been prototyped (de Souza et al. (2008)) within the scope of the SOCRADES project. However today web services on devices, have some non negligible requirements on memory, computational power and storage. It might be that these do not pose a barrier in the future, however they pose a barrier in the short term and for devices with older technologies that can still be found on the majority of shop-floors. As such a solution needs to be found, since the lifecycle of such infrastructures in some domains can be decades, and business continuity must be guaranteed.

As pointed out not all devices will have the resources to run web services natively. Even if that was feasible it might not really be reasonable (or offer any business advantages) in some cases as task-specific devices may assume a very specialized role, that can be fully accomplished with no advanced capabilities (e.g. a proximity sensor, and RFID tag etc). Generally we can see different device categories in the future factory:

- *Passive non-electronic devices*: These are devices physically not capable of WS-technology as they do not feature any electronic capability Examples include bolts, buckets etc. These passive devices can be monitored indirectly, e.g. via RFID or barcodes. The tags themselves (e.g. a wireless sensor) or wrappers around the attached tags could make it possible for such devices to participate in the future factory via simple services.
- *Resource constrained devices*: These devices have the power to communicate and process information. However, their resources are so limited that it would not be feasible or reasonable to deploy WS on them. In this situation, however, it seems very rewarding to connect them to wrapper/Gateway and/or service Mediator devices that encapsulate the device functionality and offer WS to the outside world (as depicted in Fig. 5). Typical such devices are today the RFID tags where the RFID reader is used as a Gateway depicting full-WS capabilities. Other lightweight alternatives might offer better integration for these devices also such as the REST approach (Liu and Connelly (2008)).
- *WS-capable devices*: These devices have enough computing resources to retrieve, store, compute, and transmit information and can stand-alone participate in the future factory infrastructure.

For all non-WS enabled devices, we can also distinguish several cases. Most of them today make it hard to wrap their functionality, due to the fact that they use proprietary communication protocols and/or low-level exchange mechanisms, typically for reading and writing individual device variables. Furthermore most of them are not networked and even when they are, they cannot interoperate and require specific landscapes to function. As technology evolves, this limitation may vanish, but it can be expected that in some cases this will not happen in the foreseeable future.

Service-enabling via functionality wrapping of legacy devices will contribute to creating the network effect necessary for gradually enlarging the application sphere of service orientation in industrial automation.

3. INTEGRATION OF DEVICES

3.1 Integration Options

The goal is to be capable of integrating any device, without taking direct consideration of its capabilities, resources or device-dependent functionality. As such, this abstract integration of devices within the SOCRADES infrastructure (Karnouskos et al. (2007)) can be realized generally in three different ways.

- Case #1: We use existing approaches for connecting to legacy devices, and then we wrap at middleware or ERP level part their functionality and offer it as a service.
- Case #2: The devices run web services natively e.g. DPWS, OPC-UA etc

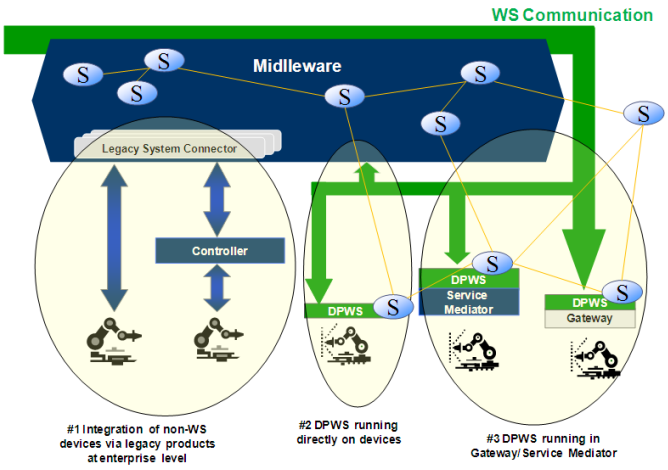

Figure 3. Options for integrating devices. DPWS is depicted as an example of implementing web services on devices

- Case #3: The devices' functionalities are wrapped by a WS enabled device one or more layers above e.g. Gateway or a Service Mediator. The communication with the service Mediator or Gateway can be done with legacy/proprietary protocols, and is managed by the wrapping device e.g. the Gateway.

Case #2 represents the main SOCRADES project vision and makes the integration straightforward. Cases #1 and Case #3 represent the alternatives when devices cannot offer WS interfaces. The former method basically consists integrating the legacy software that controls the device in middleware. In this scheme the middleware connects directly to the legacy system connectors using a predefined (possibly proprietary) protocol(s).

To realise Case #3 two concepts are proposed i.e. that of a Gateway and the Service Mediator approach (as depicted in Figure 5).

- Gateway: In this approach, the non-WS enabled device is hidden by an intelligent proxy called a Gateway. Such a component basically understands the device's proprietary protocol and exposes a WS interface. As such, the device's functionality is wrapped and now it can be addressed as any other service - whilst the fact that the device it represents is not WS-enabled remains hidden. In this approach usually the offered service(s) are bound to one specific device's functionality; something that differentiates it from the Service Mediator.
- Service Mediator: Although the Gateway mainly focuses on explicit devices, the Service Mediator focuses on services independent of the number of devices needed to support the specific service. As such it features internal functionality composition engine that aggregates the device data, possibly computes on them and mediates the resulted functionality to the prospective clients by offering a WS end-point. The Service Mediator offers service functionality that may be complex and e.g. the result of combination of info from several devices which it controls or can communicate with.

3.2 Service enabled devices

Business applications will need to access device data and state preferably always through (web) services. Although other connection operations exist, only the (web) service abstraction delivers a message-oriented communication method truly independent from the underlying operating system and programming language. The basic need is to find at which level of the architecture those complex services are provided in a form that is ready for consumption by the enterprise system. The issue should not be discussed only bottom-up, i.e. derived from the level of functionality that automation devices can offer, but primarily top-down, i.e. defined by the data exchanged between the enterprise software and shop floor. The enterprise system needs direct access to timely and context specific information (de Souza et al. (2008)). As such many unnecessary details should be hidden and an abstract service should capture only the desired functionality. The last is possible usually as a composition of other more generic services.

3.3 Legacy devices

SOCRADES is introducing a novel kind of device access and integration paradigm. Replacement of legacy infrastructure will come gradually and therefore transition approaches need to be defined (Bangemann et al. (2008)). In parallel we must guarantee coexistence of legacy and future infrastructure as well as minimization of downtimes and media breaks. As such the only viable option is an evolutionary approach, where non-WS-enabled devices are software-updated to include a web service stack (if technically feasible), or be replaced one after the other, because they have reached the end of their lifetime, or new functionality on the mechanical level is required. In this way, WS-enabled devices can gradually replace the conventional systems of today.

To have the two classes of systems, the SOA-enabled and the conventional ones, communicate with each other the new devices could come with a dual interface, providing both WS services as well as some other (e.g. vendor-specific/proprietary) protocol. This option is very useful for the transition phase. As soon as a significant part of the production environment has these dual-stack interfaces, the whole system could be re-configured and service interaction could be used as the single interaction method between devices. This switch to SOA-based control requires the remaining non-WS devices to be integrated in the new system. The preferred way of integration is to have proxy WS devices and services for the real devices. As a rule of thumb, the proxies should be as close as possible (in terms of network distance) to the real devices and should be instantiated at the lowest possible layer of the system.

Figure 4 depicts a maximum hierarchy of devices on the shop floor. All of these components are optional and it is rather unlikely that they all are present in one installation. What we commonly find is rather subsets of these systems. In this maximum end-to-end setup, a sensor e.g. an object presence sensor, that either has an analogue/digital, serial output will be at the bottom layer. This sensor is connected to a PLC which in turn connects to a single-board computer (SBC), that is conducting some

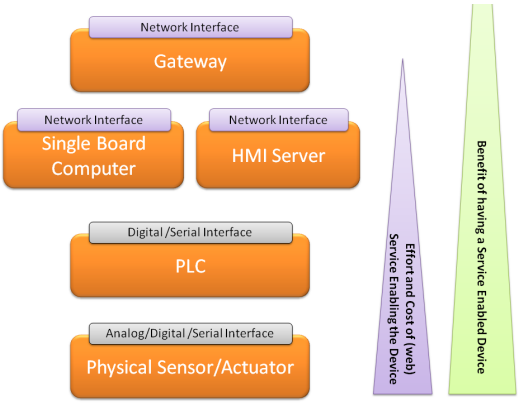

Figure 4. Shop-floor device hierarchy and estimated effort/benefit of adding web services

higher level control and/or an HMI server that creates some Graphical User Interface (GUI) for the production line operator. On top of these components, an additional Gateway might expose a higher-level interface e.g. via web services.

Generally, the lower the level at which a non-WS-enabled device is wrapped into a WS-compliant web service, the more flexibly it can participate in the device SOA compositions. The lowest feasible level would be the PLC (assuming of course that no network adapter is on the device) that could have in addition to its cyclic, real-time, control part a second part that hosts the WS stack (or just can speak *http* protocol in case of REST). This very low-level integration is however very costly and would require to re-design or introduce significant changes in PLC devices. Therefore it is practically preferable to add WS support using single-board computers or complete Gateways implemented on industrial PCs.

4. GATEWAY VS. SERVICE MEDIATOR

As we have pointed out in previous sections, the Gateway and the Service Mediator are seen as viable approaches (as depicted in Fig. 5) that would bridge the gap between the present and the future fully SOA enabled factory shop-floor.

4.1 The Gateway

A Gateway is a device that controls a set of lower-level non-service-enabled devices, each of which is exposed by the Gateway as a service-enabled device. This approach allows to gradually replace limited-resource devices or legacy devices by natively WS-enabled devices without impacting the applications using these devices. This is possible since the same web service interface is offered this time by the WS-enabled device and not by the Gateway. This approach is used when each of the controlled devices needs to be known and addressed individually by higher-level services or applications.

The Gateway approach requires some specific support from a DPWS implementation. Indeed, while a standard DPWS-enabled device is only required to store and manage its own discovery, description and hosted services metadata, a Gateway needs to support a multitude of

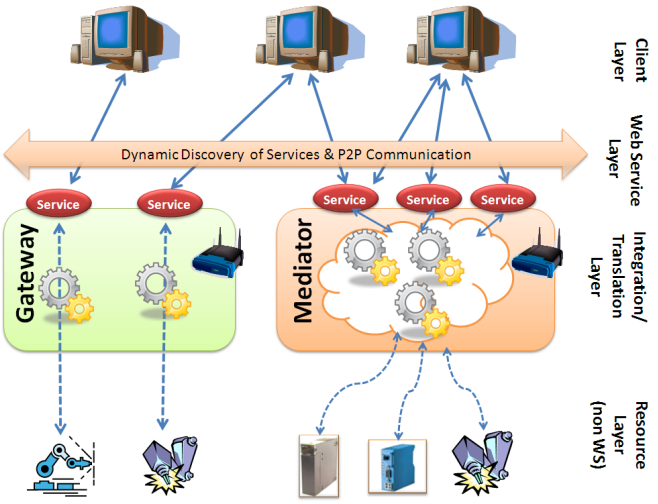

Figure 5. Non-service enabled device integration: Gateway vs. Service Mediator concepts

devices. It is therefore necessary to introduce a registry for devices and hosted services that helps structure and manage the required information. When several instances of the same device type are present, the registry distinguishes between class- and instance-level information, both for devices and hosted services, so as to factor the information common to all instances and thus to save also memory.

4.2 The Service Mediator

Originally meant to aggregate various data sources (e.g. databases, log files, etc.), the Mediator components evolved and are now used to not only aggregate various services but possibly also compute/process the data they acquire before exposing it as a service. Service Mediators aggregate, manage and eventually represent services based on some semantics (e.g. using ontologies).

In our case the Service Mediator could be used to aggregate various non WS-enabled devices. This way, higher level application could communicate to Service Mediators offering WS, instead of communicating to devices with proprietary interfaces. The benefits are clear, as we don't have the hassle of (proprietary) driver integration. Furthermore now processing of data can be done at Service Mediator level and more complex behavior can be created, that was not possible before from the standalone devices.

4.3 Discussion

Service Mediators can be used instead of simple Gateways whenever we want to introduce some low-level semantics and multiplex functionality. Consider, for example a wireless sensor network monitoring temperature along a conveyor belt (shop floor). Such a network can be composed of thousands of temperature sensors yet, the interesting service on the top floor is not the services offered by each and every sensor but rather the average temperature on the conveyor belt. A first prototype demonstrating this concept has been realized (Savio et al. (2008)).

As shown on Fig. 5 using Service Mediators introduces another level of abstraction and aggregation in between

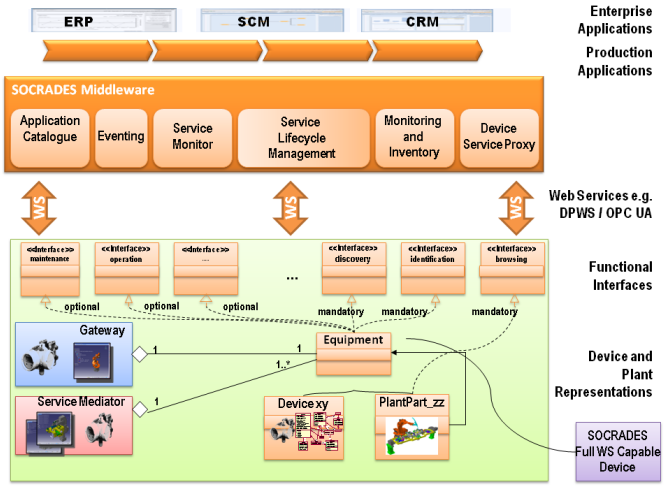

Figure 6. Mapping of fieldbus profiles to web services

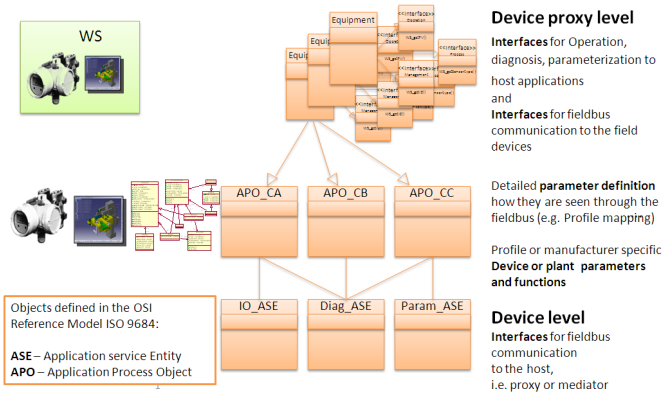

Figure 7. Mapping of fieldbus profiles to web services by means of ISO/ OSI-Reference Model: Using ASEs and APOs

the clients and devices. Thus, seen from the outside, there might not be significant difference between a Service Mediator and a composite service that relies on a set of service-enabled devices.

A Service Mediator is a device that controls a set of lower-level non-service-enabled devices, which it uses to implement a process of which it exposes a service interface. Thus the individual lower-level devices are invisible outside of the Mediator. On the contrary the Gateway depicts as a service functionality that can be directly related to a specific device - as such once can directly relate and identify an explicit device as the source of service offered by the Gateway.

Both the Gateway and the Service Mediator can host several services - limited only by their internal resources. Both approaches enable the functionality of the shop floor to be more accessible and tap into an event based infrastructure where devices indirectly (via their proxy) and functionality can be dynamically discovered (Edwards (2006)) and used e.g. due to the WS-Eventing support of DPWS.

5. IMPLEMENTING THE SERVICE-ENABLED LEGACY INFRASTRUCTURE

In the context of the SOCRADES project we have developed both the Service Mediator and Gateway approaches, that connect via web services to the also developed middleware as depicted in Fig. 6.

The term equipment is used as an abstraction of anything, which commonly identifies measurement and actuation means, as well as plant or plant parts, which represent mechatronic technological units. Both equipment classes, device and plant part, have descriptions containing meta-information, which are necessary to convert their data to WS messages. The Gateways and Mediators have access to device and plant data as well as to the associated metadata.

Devices and mechatronic units consist of mechanical, electronic, electrical, and sometimes hydraulic and software parts. Fig. 6 shows at the bottom how a plant part aggregates equipment and the specialization of equipment to device and plant part. Fig. 7 shows an information model for equipment as class diagram. This can be a device model as defined by field buses, the resource model of OPC UA or the equipment model of the CAMX/IPC 25xx family. These models cover structural and on-line data aspects. However existing and standardized models should be used as far as possible because the majority of legacy systems conforms to them.

An equipment provides a collection of web services, which offer its functionalities to the applications such as control, maintenance, production and quality.Some of these services implement some mandatory for SOCRADES equipment functionality in order to guarantee cooperation with the enterprise applications.

It is assumed that Gateways and Service Mediators for legacy systems are connected to an industrial communication system, e.g. Modbus, PROFIBUS/PROFINET, CAN etc. Therefore the equipment model is extended by its communication means for internal data transport. This aspect is hidden when building a DPWS or OPC UA functionality within the Gateway or Service Mediator. The communication model part of the equipment is introduced for an unambiguous way to integrate field buses into the Gateway / Service Mediator.

Most of the market relevant field buses (including the Ethernet based) relevant in the application domain of SOCRADES are standardized by IEC or ISO. The IEC 61158 / IEC 61784 series contains approx. 20 different bus systems. Common for all is the way of specifying the communication services and protocols using the ISO OSI reference model ISO 9684. This model defines also how to describe the communication integration in applications. Furthermore, most today's devices and field bus profiles are based on a mixture out of informal (i.e. textual) and state machine models (also known as parameter list model IEC 62390). Additionally there is a list of all variables and parameters which can be grouped to functional or structural units. For configuration and parameterization purposes there are description languages such as Electronic Device Description Language (EDDL IEC 61804-3), Electronic Data Sheet (EDS ISO 15745) or DDXML (ISO 15745). Most Devices are already purchased with files containing their descriptions so that commissioning tools get the appropriate information.

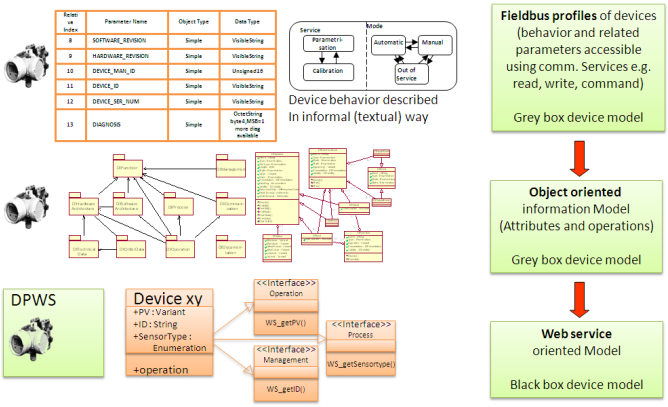

Figure 8. Mapping of fieldbus profiles to web services - abstraction hierarchies

Some fieldbus organizations provide their device class or application function descriptions based on object oriented models. The variables and parameters are accessible using the object operations. The object, i.e. visible device functions are part of the object behavior description. To assimilate this information a web service representation of the equipment behavior and data must be realized. For this purpose the mappings between the parameter list model and the object model to web service model are needed, as depicted in Fig. 8.

6. CONCLUSION

It is clear that we are heading towards an infrastructure where heterogeneity will be dominant and not all devices will have the capability of hosting tapping directly to the future SOA dominated shop floor by implementing WS natively and provide their functionality as a service to the others. In fact, the last might not only be infeasible due to technological constraints, but it also might not make sense from the business point of view. Therefore any approach proposed for the future manufacturing domain, has to make sure that all types of devices can be directly and indirectly integrated in a global communication infrastructure.

To ease legacy infrastructure transition to the SOA shop-floor we have shown how this can be realised via a Gateway or a Service Mediator approach. The overall architecture and its components within the SOCRADES project have been designed while taking into account a wide requirements set, as well as the existing service bridging concepts and technologies. The concepts described here have been prototyped and we are planning for a real-world evaluation in trials within 2009 in the scope of the SOCRADES project.

ACKNOWLEDGEMENTS

The authors would like to thank the European Commission and the partners of the European IST FP6 project "Service-Oriented Cross-layer infRAstructure for Distributed smart Embedded devices" (SOCRADES - `www.socrades.eu`), for their support.

REFERENCES

(2008). Soa in manufacturing guidebook. URL `ftp://ftp.software.ibm.com/software/applications/plm/resources/MESA_SOAinManufacturingGuidebook.pdf`.

(2008). Toward a european strategy for the future internet. URL `http://www.europeansoftware.org/documents/SAP_WP_FutureInternet.pdf`.

Bangemann, T., Diedrich, C., Colombo, A.W., and Karnouskos, S. (2008). Socrades - service oriented architecture in der automatisierungstechnik. In *Automation 2008, 3-4 June 2008, Baden-Baden, Germany.* (in German).

Chan, S., Kaler, C., Kuehnel, T., Regnier, A., Roe, B., Sather, D., Schlimmer, J., Sekine, H., Walter, D., Weast, J., Whitehead, D., and Wright, D. (2005). Devices profile for web services. Microsoft Developers Network Library.

Colombo, A.W. and Karnouskos, S. (2009). Towards the factory of the future - a service-oriented cross-layer infrastructure. *in the book ICT Shaping the World, A Scientific View, ETSI, John Wiley and Sons Ltd*, ISBN: 9780470741306.

de Souza, L.M.S., Spiess, P., Guinard, D., Koehler, M., Karnouskos, S., and Savio, D. (2008). Socrades: A web service based shop floor integration infrastructure. In *Proc. of the Internet of Things (IOT 2008)*. Springer.

Edwards, W.K. (2006). Discovery systems in ubiquitous computing. *IEEE Pervasive Computing*, 5(2), 70–77. doi:http://doi.ieeecomputersociety.org/10.1109/MPRV.2006.28.

Fleisch, E. and Mattern, F. (2005). *Das Internet der Dinge*. Springer.

Hannelius, T., Salmenpera, M., and Kuikka, S. (2008). Roadmap to adopting opc ua. In *Proc. 6th IEEE International Conference on Industrial Informatics INDIN 2008*, 756–761. doi:10.1109/INDIN.2008.4618203.

Karnouskos, S., Baecker, O., de Souza, L.M.S., and Spiess, P. (2007). Integration of soa-ready networked embedded devices in enterprise systems via a cross-layered web service infrastructure. In *Proc. ETFA Emerging Technologies & Factory Automation IEEE Conference on*, 293–300. doi:10.1109/EFTA.2007.4416781.

Liu, Y. and Connelly, K. (2008). Realizing an open ubiquitous environment in a restful way. In *Proc. IEEE International Conference on Web Services ICWS '08*, 96–103. doi:10.1109/ICWS.2008.64.

Ma, X. and Luo, W. (2008). The analysis of 6lowpan technology. In *Proc. Pacific-Asia Workshop on Computational Intelligence and Industrial Application PACIIA '08*, volume 1, 963–966. doi:10.1109/PACIIA.2008.72.

Savio, D., Karnouskos, S., Wuwer, D., and Bangemann, T. (2008). Dynamically optimized production planning using cross-layer soa. In *Proc. 32nd Annual IEEE International Computer Software and Applications COMPSAC '08*, 1361–1365. doi:10.1109/COMPSAC.2008.219.

Spiess, P. and Karnouskos, S. (2007). Maximizing the business value of networked embedded systems through process-level integration into enterprise software. In *Proc. 2nd International Conference on Pervasive Computing and Applications ICPCA 2007*, 536–541. doi:10.1109/ICPCA.2007.4365502.