Web of Things Technologies for Embedded Applications

Jörg Heuer, Daniel Peintner, Sebastian Käbisch, Johannes Hund, Darko Anicic
Siemens AG,Corporate Technology,Munich,Germany
{joerg.heuer, daniel.peintner.ext, sebastian.kaebisch, johannes.hund, darko.anicic}@siemens.com

Keywords: Embedded Systems, Web services, CoAP, EXI, Web Semantics, Efficient RDF, XMPP

Abstract Web of things beyond controlling embedded devices with smart phones – this contribution raises the question what motivates to make embedded devices a full citizen of the web, what is required to integrate them and provides examples for relevant technologies which starts to enable such an integration. Concretely we discuss based on the application domains of smart home and smart grid motivation, use cases and requirements for a web integration. To stimulate a discussion on the different means of integration we evaluate some web technologies which already today enable such integration and share thoughts about the balance between adaption of web technology for the use in the embedded domain and the integration into today's web.

1 INTRODUCTION

No doubt, "things" have become smart in the last decade: take as an example the water pump for central heating which tries to estimate the thermostat settings in the home. However it does it by local measuring pressure differences without explicit online information exchange between the other "things" such as thermostats or the heater control in the home. Isolated processes like these could however benefit from information exchange with other "things" respective. One way would of course be defining specific communicative means to exchange information between e.g. a water pump and thermostats or heater controls. But then how to extend it to the next "thing", e.g. a solar heating or the weather forecast?

Isn't this somewhat similar what we faced earlier last century as a challenge where the information exchange between persons and processes was time consuming? We tried to solve and optimize issues based on the local knowledge. With web technologies means are provided to simplify information sharing, structuring, description, indexing, service implementation and hosting. By this we managed to deal with our daily tasks significantly more efficient by integrating different information sources e.g., least cost navigation to a patrol station.

So, even though it is about "things" and not persons, the questions which come up to mind are:

- *Can we do it quite similar to the web?* Section 2 will address this question by considering two use cases and their requirements.
- *Where do we need to adapt because "things" are different?*
  This question is discussed in Section 3 by considering a selection of today's web technologies.
- *Can we do it still as part of the web?* Section 4 will get back to this questioning.
- *What hinders us to do it already today?* We hope to discuss a common understanding by means of the workshop.

2 USE CASES & REQUIREMENTS

Can we do it quite similar? – To discuss this question we picked in this section two example domains of "things" with smart home and smart grid to evaluate their requirements and challenges and whether they are comparable to Web scenarios.

2.1 Use Cases

2.1.1 Smart Home

Today's modern home is already enriched with a set of heterogeneous device equipments (the "things") from different vendors including washing machine, alarm systems (e.g., fire), inverters, water pumps, etc.. Typically, smart home scenarios address applications around smart controlling and energy savings, as they are also discussed in similar manner for buildings [pinta]. However, how can this basically be achieved?

A first fundamental step could be to have a network-enabled device landscape at home where we have accessible device data via well known interfaces. This opens the opportunity that devices can be harmonized with each other, e.g., by setting up operation parameters in optimized manner based on the desired applications. Furthermore, for sharing experiences such as malfunctions or further optimization potentials, home owners may find someone else who has a similar home configuration footprint, e.g., via the web.

Having a "networking of things at home" that is based on heterogeneous devices, even from different application domains (e.g., smart grid), as well as the consideration of different hardware resource classes down to small microcontrollers, requests a set of standard technologies. This can then also involve the potentials that come from the "networking of processes", e.g., as described with the water pump in the introduction where users normally take benefit from a process running in the background but are not aware of it. We should discuss and ask the questions, whether well-known standardized web technologies such as web services and semantic web are suitable and applicable for all the smart home applications, which also involve the usage of different kind of embedded device resource classes.

2.1.2 Smart Grid

In the application domain of smart grid, "things" such as distributed energy resources (DERs), e.g. smart electrical equipment of grid operators or devices and households with smart energy management will be a major use case of WoT technologies [SmartGridCom].

Deployments in large numbers and the demand for interoperation and information exchange across several stakeholders and long product lifecycles require standardized protocols and interoperable, maintainable technologies. Cost factors and wide area of service advise the usage of public networks instead of dedicated lines. Security and privacy play a key role in smart grid regulations, enforcing state-of-the-art-encryption and ensured privacy also over public networks. Applications in the field of smart grid need to be portable and hardware-independent based on standardized APIs, services and data models. Wouldn't that be a good fit for well-known web technologies which enable the information exchange?

However, especially in the distribution grid, cost factors demand the usage of devices and controllers with restricted resources, which, due to the wide spatial area of service, need to enable decades of unattended operation. Since low footprint hardware prohibits the direct use of web protocols, how should they interact?

While this setup greatly differs from the common modus operandi of nowadays web applications, adaptations of well-known web technologies will enable their seamless usage on resource-restricted embedded devices. So, we should discuss how the same standardized APIs, services and data models can be used across a broad spectrum of devices and processing powers, enabling seamless communication and applications. How can well-known web technologies be adapted to run on low-cost hardware?

. 2.2 Requirements

WoT could be interpreted as approach to apply web technologies to the network of "things" or as a way to make every physical object ("thing") a first class citizen of the World Wide Web. In both cases several requirements are to be met. Already by considering the initial example of a water pump one can spot many requirements. Find below a non-exhaustive list of requirements.

- *Observation* – Scalability and long lasting communication relationship require concepts of observation, where a state change is proactively propagated.
- *Unsupervised operation* – Contrary to Web applications, the "things" are usually operating without any human interaction.
- *Efficiency* – Data exchange should be efficient with regards to memory,

bandwidth, limited processing power, cost, and energy.

- *Domain mix* – The data exchange in the WoT ought to deal with very different domains ranging from public networks to closed domains and also across domains.
- *Openness vs. security* – The overall system needs to be as open and as extendable as possible but must not disregard security aspects such as a role-based access control.
- *Self** – Easy setup including self-configuration and self-description enhance self-healing and ease future adaptations.
- *Interoperability* – Data needs to be comprehended unambiguously by both human users and software programs across different platforms and domains. It offers interaction between heterogeneous things, machines, and smart objects on a higher level of abstraction. This is a prerequisite for the creation of WoT value added services and applications.
- *Interpretation of data and knowledge* – The data generated by WoT devices needs to be understandable by machines and humans, without prior knowledge about devices that produced them.
- *Unambiguity* – We need unambiguous meaning of data and properties. For example, it is not sufficient to know that there exists a device, but it is important to know what exactly a capability of the device is, and to unambiguously understand the data it produces or consumes. To this category also belongs the object or entity abstraction, which describes common functions of a device in an abstract way, independent from different vendors or standards
- *Data/knowledge integration* – Data and knowledge from WoT devices and applications need to integrate with multiple external and/or internal sources (in a heterogeneous environment). In mesh-up applications, it also helps in understanding the original data (before the mesh-up is established) and enables applications and services in a secondary WoT market;
- *Engineering and management of WoT applications, including auto-configuration and re-configuration capability* – Massively distributed systems such as WoT systems need to have interfaces and their properties/capabilities described in a machine readable form in order to support engineering and management of themselves, e.g., to enable the plug-and-play functionality.
- *Timeliness of data* – The physical world is changing fast, and WoT applications that aim to capture those changes, process them and react to them are dynamic. Therefore adaptive and event-driven processes are norm in the WoT. One of the main benefits of the WoT integration is that processes become more adaptive to what is actually happening in the real world. Inherently, this is based on events that are either detected directly or by real-time analysis of sensor data. Such events can occur at any time in WoT-related processes, they need to be analyzed in the timely fashion, and often necessary reactions need to be figured out and taken on-the-fly.

3 DISCUSSIONS OF WOT TECHNOLOGIES

Where do we need to adapt? – Already today web technologies are pragmatically used in “things” such as e.g., water pumps at home to configure those via smart phones. To not only limit the use to isolated applications but address the broader use sketched out in the previous section we are discussing where adaptation of web technologies can help to fulfil the previously stated requirements.

3.1 Efficient XML Interchange (EXI)

In recent years the need for supporting semi-structured data exchange in heterogeneous application areas has been raised due to the tremendous increase in communication devices. In most of the cases the Extensible Markup Language (XML) [w3cxml] provided an attractive solution due to its high acceptance in the community and its flexibility. However, despite its success, XML is text-based and tends to be verbose and hardly processable on limited microcontrollers. Are there more efficient representations of XML?

Many so called *binary* XML formats were developed in the past to overcome the problems that have been identified with regard to XML in restricted environments. The Efficient XML Interchange (EXI) format [w3cexi] is such a promising compact representation of the XML

Information Set [w3cxis] produced by the W3C. It is intended to be the last binary XML format by simultaneously optimizing performance and the utilization of computational resources. The EXI format uses a relatively simple grammar-driven approach that achieves very efficient encodings (EXI streams) for a broad range of use-cases. Due to a straightforward encoding algorithm and a small set of data types, EXI processors can be implemented on devices with limited capacity. Besides other relevant properties such as encodings with and even without XML schema information, as well as schema deviations or partial schemas, the EXI format offers a variety of additional useful features. As such an EXI Profile for limiting usage of dynamic memory [limw3cexi] has been elaborated intended for low-resource or ultra-constrained devices. Such devices lack run-time memory allocation capabilities or at best have extremely limited dynamic memory resources. In [kphk2011] it has been shown that EXI can be deployed on very limited microcontrollers and by doing so it allows such limited devices to seamlessly interact with the *traditional* Web.

3.2 CoAP

HTTP is the backbone and unified transport layer of the Web. However, being based on textual representation and TCP transport, it cannot be used for controllers with restricted resources. An open question is therefore: Which will be the equivalent and seamlessly integrated HTTP adaption that is more feasible to the embedded domain?

The constrained application protocol (CoAP) [ietf-coap] is a direct translation of HTTP for embedded devices will mitigate the shortcomings that prohibit the use of HTTP on embedded devices. CoAP is enabling the direct usage of web-technologies for resource-constrained devices using UDP transport and binary representation of HTTP. Besides the CoAP core specification there exists an extension to realize subscription and notification based mechanisms, bypassing the rigid request-response pattern: CoAP Observe [coap-observe]. The main idea is that a client is able to observe a resource that is provided by a CoAP server. Thereupon, over a period of time the server proactively notifies the client when the state of the resource changes. This bandwidth-friendly approach would realize the event based interactions that can be found in today's embedded applications without resorting to methods like polling.

3.3 XMPP

Communication is intrinsically unreliable by nature, especially wireless communication and in rural areas. So how can we achieve constant availability, ensured delivery and efficient bandwidth usage? Can we communicate with web technologies even between resource-constrained devices, which are connected via lossy links and network situation such as residential DSL lines?

One approach to achieve this is to address compensations in application layer protocols. Nowadays web technology offers solutions for example in chat applications.

Originating from the open chat protocol "jabber", XMPP is a W3C-standardized protocol offering flexible real-time messaging and generalized routing for arbitrary XML payloads [xsf1]. Through an own standardization process, the protocol offers numerous extensions, for example communication patterns like publish-subscribe. As the protocol is already recognized as a potential candidate for WoT, several extensions to adapt it for these use cases do exist. But how can it be adapted to the stringent efficiency requirements? Since the entire protocol is based on XML, EXI can be used to increase efficiency and enable also embedded devices to be connected via XMPP [xep0322].

This offers the possibility to exchange M2M-information for different applications between heterogeneous devices even over unreliable links and difficult network situations such as NAT gateways. But what other features of a chat protocol could also be beneficial? Do in addition to WoT-specific extensions, also concepts from the chat domain like presence awareness, domain federation, service discovery and TLS-based encryption by default make XMPP a suitable candidate for communications in a Web of Things?

3.4 Semantics for the Web of Things

The increasing number of heterogeneous and interoperable devices in WoT yields to more complex embedded networks. Semantic Web technologies are seen as a key enabler of the interoperability in WoT applications. In particular, these technologies have potential to enable WoT devices to discover other devices, based on their capabilities; to help in engineering and maintenance of WoT devices in large systems; to provide intelligence to WoT devices (e.g., transforming low-level WoT data to high-level knowledge required for acting in physical world) and so forth.

However, before semantics may bring benefits to embedded applications in the domain of WoT, we need to enable semantics to be processed by embedded devices. What is needed for *Embedded Semantics*?

- A compact representation of embedded semantic data;
- Processing and analyzing of embedded semantic data;
- Ontologies and semantic models for embedded application in the context of WoT?

In following subsection these three aspects of embedded semantics are discussed.

3.4.1 Efficient RDF

Increasing numbers of interchanging heterogeneous devices yield more and more complex embedded networks in the future. To face this challenge, embedded semantics could be used to support the engineering process or to realize a plug-and-play integration of new devices in an embedded network at operation time. A semantic repository such as a RDF store on an embedded device can be used to save, update, delete, and search semantically relevant data. In general, traditional semantic representation such as known from Semantic Web is not feasible to microcontrollers with limited hardware resources (e.g., memory, processing, and bandwidth) due to the textual representation like standard RDF with plain-text XML. How can we overcome this issue? EXI as already presented in Section 3.1 is a well known approach to overcome in a standardized manner the textual encoding and operates in a high efficient and compact way that is also suitable for microcontrollers. Thus, it would make sense to operate with a semantic repository such as a RDF store based on the mechanism of EXI.

3.4.2 Processing and analyzing Embedded Semantics

In comparison to embedded devices, current mechanisms for processing data in the Semantic Web are mainly suited for machines with greater computational resources. Furthermore, they are tailored for processing time-invariant or slowly evolving semantic data. The WoT data is often generated as streams, and applications require continuous asynchronous processing of semantic data streams.

Processing of semantic data in an embedded environment therefore demands revision of concepts, architectures, and algorithms, commonly used in the Semantic Web, in order to be applicable for constrained devices. Embedded devices have significantly lower CPU and memory capabilities, as well as limited network throughput and power resources.

Further on, regarding the processing of time-varying semantic data two types of processing are important: *querying* and *reasoning*.

SPARQL, a W3C standard for querying data in RDF format [w3csparql], has already seen modifications in direction of querying data streams, see for example [exec-sparql], [proc-link], [event-proc]. Moreover W3C has recently imitated *RDF Stream Processing Community Group* (RSP) addressing this topic [rsp]. This work might also be concerned for applications in the embedded domain.

However WoT applications will certainly demand not only querying semantic streams, but also creating high-level abstractions where sensory observation data, enriched with contextual knowledge, is used in a logical inference process to derive perceptions that cannot be derived solely from the raw observations. Logic inference over semantic streaming data and background knowledge is known also as *Stream Reasoning* [stream-reason]. The high-level abstractions, in relation to domain knowledge in different applications, can create a source of perception which will be the driving asset for developing intelligent applications and smart environments that use the WoT data [sem-iot]. The challenge, how to enable Stream Reasoning on constrained devices, remains an interesting research topic.

3.4.3 Ontologies and semantic models

In semantic web applications, the main vehicle to give information well-defined meaning is realized with ontologies. They unambiguously define meaning of information, are interpretable by both humans and machines, are defined on commonly accepted understanding of a certain domain (e.g., smart home domain) and are shared. Moreover they enable machines to process ontologically represented knowledge and to reason about it.

Ontologies and semantic models are important for success of semantic WoT applications. They are important for embedded WoT applications too. In this respect, for example, the work on Semantic

Sensor Network (SSN) ontology, from the W3C Incubator group SSN-XG [ssnxg], is of interest. However, apart from cross-domain ontologies, it is of paramount importance to have domain-specific ontologies also available. For instance, see BACnet ontology [bacowl] as an open source attempt to formalize important aspects of BACnet - a data communication protocol for Building Automation and control networks [ashare]. Similar approaches seem desirable in the future for other domains too (e.g., smart grids, smart factory, smart home, smart city etc.). Finally, for WoT applications from embedded domain, it is additionally important that these ontologies and semantic models are represented in compact formats (see Section 3.4.1) and processable by constrained devices (see Section 3.4.2).

3.5 Embedded Service Containers

Service technologies are an important building block of nowadays Web applications, however there is currently still a conceptual barrier between IT services and the system architecture on resource-constrained embedded devices. How could the concept of service architectures be ported to the embedded domain?

Obviously, a powerful runtime platform is needed. Similar to container frameworks such as J2EE [j2ee-spec], WoT applications or services could consist of semantically enriched code written in a common programming language that interfaces with standardized platform services. The applications might run in a sandboxed environment that shields the application from external influences, including other applications, while a service-oriented platform API provides conflict-free access to hardware devices such as sensors and actuators.

Scheduling and resource management functionality provided by the platform guarantees realtime performance. Semantic annotations might simplify the search for and orchestration of services in large scale networks.

With a technology like this, the interoperability and scalability known from classical web applications can be ported to the embedded domain, allowing Things to participate and interact at eye level with IT services in the upcoming Web of Things.

4 WEB INTEGRATION

Can we still do it as part of the web? – In the last section we raised the question what are adapted web technologies to implement WoT. However the second interpretation of WoT – make every physical object ("thing") a first class citizen of the World Wide Web – requires that also the adapted web technologies interfaces with todays web. These goals might be in conflict to each other.

To avoid the risk of again diverging goals it seems to be important to discuss generic mappings of adapted web technologies to the existing web deployments. For instance it has been a requirement of EXI to provide a generic binary representation of the XML Infoset, so that the same information is conveyed, regardless if it is represented in EXI or XML. Another example is the ability of direct translation between HTTP and CoAP. By approaches like these, the specific WoT requirements can be fulfilled while still maintaining the ability of web integration.

5 EXPECTATIONS ON THE WOT WORKSHOP

With this input contribution we would like to contribute to the following goals in the W3C Workshop

- A common understanding of the WoT domain
- Evaluation of WoT related activities
- Identification of potential work topics for W3C
- Setup of a WoT technology landscape

REFERENCES

[w3cexi] John Schneider, Takuki Kamiya, Daniel Peintner, and Rumen Kyusakov. Efficient XML Interchange (EXI) Format 1.0 (Second Edition). W3C Recommendation, W3C, February 2014. http://www.w3.org/TR/2014/REC-exi-20140211/.

[w3cxml] Tim Bray, Jean Paoli, Eve Maler, François Yergeau, and C. M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C recommendation, W3C, November 2008. http://www.w3.org/TR/2008/REC-xml-20081126/.

[w3cxis] Cowan, J. and Tobin, R. (2004). XML Information Set (Second Edition). http://www.w3.org/TR/xml-infoset/. W3C Recommendation.

[limw3cexi]Youenn Fablet and Daniel Peintner. Efficient XML Interchange (EXI) Profile for limiting usage of dynamic memory. W3C Proposed Recommendation, W3C, May 2014. http://www.w3.org/TR/2014/PR-exi-profile-20140506/.

[SmartGridCom] S. Käbisch, A. Schmitt, M. Winter, and J. Heuer: Interconnections and Communications of Electric Vehicles and Smart Grids. First IEEE International Conference on Smart Grid Communications (SmartGridComm) 2010, Gaithersburg, Maryland, USA

[xsf1] Xmpp Standards Foundation. About XMPP, http://xmpp.org/about-xmpp/

[xep0322] Peter Waher, Yusuke DOI. XEP-0322: Efficient XML Interchange (EXI) Format. http://xmpp.org/extensions/xep-0322.html

[ietf-coap] Zach Shelby, Klaus Hartke, Carsten Bormann. Constrained Application Protocol (CoAP). http://tools.ietf.org/html/draft-ietf-core-coap-18

[coap-observe] K. Hartke
Observing Resources in CoAP
http://tools.ietf.org/html/draft-ietf-core-observe-08

[j2ee-spec] Oracle America, Inc. JSR-342 Java Platform, Enterprise Edition 7 Specification http://download.oracle.com/otn-pub/jcp/java_ee-7-fr-spec/JavaEE_Platform_Spec.pdf

[kphk2011] S. Käbisch, D. Peintner, J. Heuer, and H. Kosch. Optimized XML-based Web Service Generation for Service Communication in Restricted Embedded Environments. In: Emerging Technologies and Factory Automation (ETFA) 2011, IEEE16th Conference on, 2011

[pinta] Javier Militino, Susana Alcalde Bagüés, Jelena Mitic. "Improving Energy Efficiency in Office Buildings" in KNX Scientific Conference, 2012.

[w3csparql] The W3C SPARQL Working Group, W3C, ed. 2013. SPARQL Query Language for RDF. http://www.w3.org/TR/sparql11-overview/.

[exec-sparql] Barbieri, Davide Francesco, Daniele Braga, Stefano Ceri, and Michael Grossniklaus. 2010. "An Execution Environment for C-SPARQL Queries." In Proceedings of the 13th International Conference on Extending Database Technology, 441–52. EDBT'10. ACM.

[proc-link] Le-Phuoc, Danh, Minh Dao-Tran, Josiane Xavier Parreira, and Manfred Hauswirth. 2011. "A Native and Adaptive Approach for Unified Processing of Linked Streams and Linked Data." In Proceedings of the 10th International Conference on The Semantic Web - Volume Part I, 370–88. ISWC'11. Berlin, Heidelberg: Springer-Verlag.

[event-proc] Anicic, Darko, Paul Fodor, Sebastian Rudolph, and Nenad Stojanovic. 2011. "EP-SPARQL: A Unified Language for Event Processing and Stream Reasoning." In Proceedings of the 20th International Conference on World Wide Web, 635–44. WWW'11. New York, NY, USA: ACM.

[stream-reason] Anicic, Darko. 2011. "Event Processing and Stream Reasoning with ETALIS". Dissertation/Ph.D. thesis, Karlsruhe, Germany: The Karlsruhe Institute of Technology (KIT). http://www.aifb.kit.edu/web/Phdthesis3035.

[sem-iot] Barnaghi, Payam, Wei Wang, Cory Henson, and Kerry Taylor. 2012. "Semantics for the Internet of Things: Early Progress and Back to the Future." Int. J. Semant. Web Inf. Syst. 8 (1): 1–21. doi:10.4018/jswis.2012010101.

[ashare] Standard 135-2012 - BACnet® - A Data Communication Protocol for Building Automation and Control Networks (ANSI Approved). ANSI/ASHRAE.

[rsp] W3C Community and Business Groups: RDF Stream Processing Community Group http://www.w3.org/community/rsp/

[ssnxg]