Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 57 (2016) 315 – 320

www.elsevier.com/locate/procedia

49th CIRP Conference on Manufacturing Systems (CIRP-CMS 2016)

OPC UA & Industrie 4.0 - enabling technology with high diversity and variability

Miriam Schleipen[a]*, Syed-Shiraz Gilani[b], Tino Bischoff[a], Julius Pfrommer[a]

[a]*Fraunhofer IOSB, Fraunhoferstr.1, 76131 Karlsruhe, Germany*
[b]*Fraunhofer IOSB-INA, Langenbruch 6, 32657 Lemgo, Germany*

* Miriam Schleipen. Tel.: +49-721-6091-382; fax: +49-721-6091-413. *E-mail address:* miriam.schleipen@iosb.fraunhofer.de

Abstract

Industrie 4.0 demands flexibility, adaptability, transparency and many more requirements which have to be fulfilled by Industrie 4.0 components or systems in order to achieve horizontal and vertical interoperability as well as interoperability over the lifecycle.
The present paper describes different scenarios and deployed use cases, which are all based on the OPC Unified Architecture (OPC UA), to state the role of OPC UA as enabling technology for flexible, adaptive, and transparent production.
Nevertheless, the scenarios, based on different methods, architectural approaches, hardware and software, are located in different application domains, and cover different parts of the OPC UA standard series. The application domains include quality defect tracking, monitoring and control, and code analysis.

Keywords: OPC UA; data; information; semantics; flexibility; adaptability; transparency

1. Introduction

Industrie 4.0 demands flexibility, adaptability, transparency and many more requirements which have to be fulfilled by Industrie 4.0 components or systems (see also [1]). The implementation strategy (Umsetzungsstrategie, April 2015) of the German Initiative Industrie 4.0 names several existing and usable approaches or technologies for individual aspects which are candidates for Industrie 4.0 standards. Due to the complexity of Industrie 4.0, there is no single standard (series), but the need to integrate and combine different standards from different domains which cover different aspects. OPC UA [2] was named as a good prospect for the communication aspect. The variability and flexibility of OPC UA ranges from simple process data acquisition to complex monitoring, control, and analysis. It covers security aspects and provides the ability to represent and cover semantics by its information model. Additionally, OPC UA is explicitly open for the integration and combination with other standards based on so called companion specifications. The number of collaborations between the OPC Foundation and other organizations grows, which focus on creating various companion specifications. These companion specifications range from general device descriptions (OPC UA for devices) and analyzer device models (ADI) over plant descriptions in AutomationML (AML for OPC UA) to the implementation of IEC PLCOpen programming models in OPC UA servers (OPC UA for IEC 61131-3).

The VDI (German Association of Engineers) GMA Committee 7.21 'Industrie 4.0' sub group 'terms' defines basic terms and definition for Industrie 4.0 under the lead of the Fraunhofer IOSB (see [3]). According to this definition, a CPPS (cyber-physical production system) is a CPS (cyber-physical system) which is used in the production. CPS is defined as system which connects real (physical) objects and processes with information processing (virtual) objects and processes via information networks which are open, partially global, and at any time connected. Optionally a CPS uses local or distant services, possesses man-machine interfaces and supports the dynamical adaption of the system during

doi:10.1016/j.procir.2016.11.055

runtime. To meet the defined requirements, the CPS must come with self-x properties. The components of the system therefore:

- must be able to communicate via networks and be aware of their communication partners (self-organization, context awareness),
- must possess a self-description (self-explaining) and protect this information in an adequate way (self-protection),
- must be able to configure and optimize themselves based on their available information (self-configuration, self-optimization, self-healing)

To this end semantic technologies are needed to realize building blocks for reaching this goal, which contribute to the specification of a Reference Model for Industry 4.0 System Architectures [4]. The Reference Architecture Model Industrie 4.0 (RAMI4.0, [5]) was published in 2015 and describes a possible general reference architecture for Industrie 4.0 components and Industrie 4.0 systems which can consist of CPS.

2. OPC UA in Cyber-Physical Production Systems (CPPS)

Even though OPC UA is mostly used in higher automation levels for the purpose of monitoring and control, it also increases device connectivity via standard communication in lower automation levels [6]. Therefore, OPC UA was named as candidate for communication aspects in RAMI4.0. The OPC-UA approach abstracts data from the network technology and software application, and offers a generic communication interface. It can be seen as one of the key technology for a transparent data representation/ transmission between heterogeneous system components [7].

But OPC UA does not only deal with simple data access, but also with a wide range of other aspects e.g. security, reliability, access control, alarming, or historical data. This contribution shows some possible building blocks which can be realized by means of OPC UA and which are listed in the following:

- **Separation of information model and data base structure:** hiding hierarchical data base structures behind object-oriented, full-meshed information models in OPC UA to be independent of the fixed structure of the data base.
- **Minimal OPC UA client functionality:** implementing base OPC UA client functionality for applications to hide OPC UA from developers
- **Graphical information model definition:** developing OPC UA servers together with end users (or a system integrator) based on graphical representation of OPC UA models (see [8])
- **Common info view:** data acquisition by means of OPC UA (e.g. with additional Raspberry Pi) e.g. for a management view which is accessible from different sources e.g. web interfaces, and which runs in parallel to the real-time communication
- **Aggregating OPC UA server functionality:** OPC UA server which integrates and combines different underlying servers and their information (see [9])
- **AutomationML or other description model as base/data source:** usage of external descriptions as base/data source for information models in OPC UA or for the automatic generation of process visualization with OPC UA connection to the production process (see [10, 11, 12])

To relate those building blocks to RAMI4.0, they can be assigned to the interoperability layers/dimension which are represented on the vertical axis. The listed building blocks of this contribution focus on integration, communication and mainly information layer in RAMI4.0 [5], see Fig. 1.

Today, the common information view is the most frequently used application of OPC UA in the industrial context which can be seen as 'door-opener' or starting point for future applications. This is why this building block is also present in all of the scenarios described in the following.

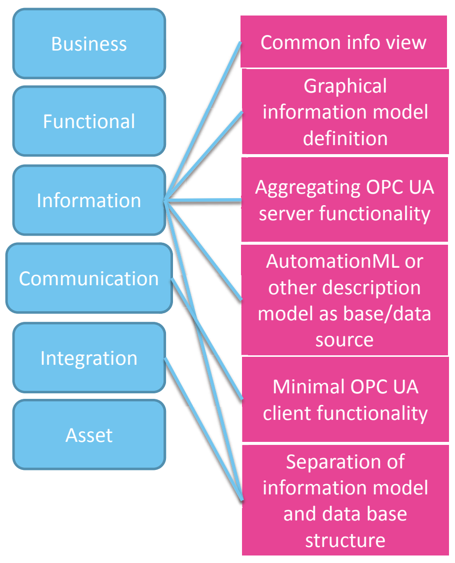
Business
Functional
Information
Communication
Integration
Asset
Common info view
Graphical information model definition
Aggregating OPC UA server functionality
AutomationML or other description model as base/data source
Minimal OPC UA client functionality
Separation of information model and data base structure

Fig. 1. Relation of OPC UA building blocks (on the right) to interoperability layers of RAMI4.0 [5] (on the left).

3. Application scenarios of OPC UA

Four different application scenarios will be described, which were designed for and realized with OPC UA and which make use of the listed building blocks. Other applications can be found e.g. in [13].

3.1. Scenario 1 – Quality defect tracking system

This application shows OPC UA to be used as an enabler for complex information processing systems in the production environment. The manual tracking of defects was time consuming as well as error prone, as operators were required to go through an extensive list of defects to find out the code, followed by manual documentation of this information. This is why these manual methods to track defects that arise during production were replaced by a system based on worker apps, management dashboards, and central information hubs with a database system (see Fig. 2).

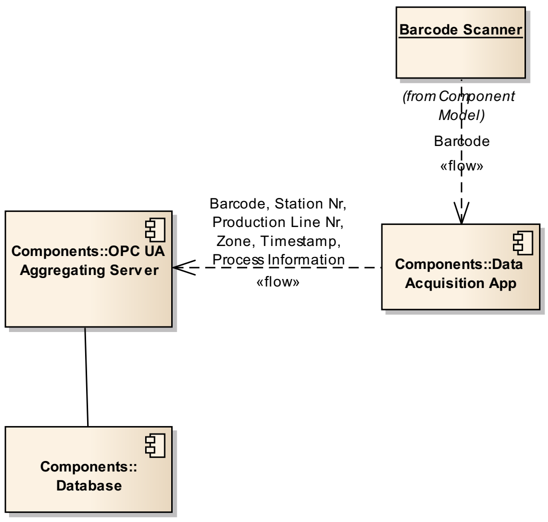
Barcode Scanner
(from Component Model)
Barcode
«flow»
Barcode, Station Nr, Production Line Nr, Zone, Timestamp, Process Information
«flow»
Components::OPC UA Aggregating Server
Components::Data Acquisition App
Components:: Database

Fig. 2. Components of OPC UA based quality defect tracking system.

All communication within the system as well as the organization of the acquired information were based on OPC UA. An OPC UA server was created, which is linked to a database, to provide and save the necessary information during the production phase. An OPC UA client, which was in the form of a windows tablet app, was implemented to communicate this information to and from OPC UA server. This tablet app was connected to a barcode scanner to simplify information acquisition by the operators as much as possible. Furthermore, an OPC UA client dashboard, which runs on any normal desktop environment, was also implemented to track the production information using several SQL queries that work with OPC UA server.

Implemented OPC UA related components in this scenario and resources used to this end were:

- OPC UA Server written in C++ based on Unified Automation C++ SDK running on Windows at least 7 / Server 2008 R2
- MS SQL-Server Database Interface for OPC UA Server written in C# based on ODBC and Microsoft .Net Framework at least Version 4 running on Windows at least 7 / Server 2008 R2
- OPC UA Client Library written in C# based on OPC Foundation Sample Code Version 1.02 and Microsoft .Net Framework at least Version 4 running on Windows at least 7 / Server 2008 R2 including a help file (CHM) for users
- Worker Station App written in C# with Microsoft .Net Framework at least Version 4.5 based on OPC UA Client Library running on Windows at least 7 / Server 2008 R2

In this application scenario, the following OPC UA building blocks were used:

- Separation of information model and data base structure
- Minimal OPC UA client functionality
- Graphical information model definition
- Common info view

3.2. Scenario 2 – Visualization of process information (monitoring)

This scenario demonstrates the ability of OPC UA to be used in safety critical systems. An OPC UA solution will be developed for a production line where production material is passed from a robot gripper to human worker. As this collaboration is safety critical, introduction of OPC UA in this production line provides the necessary information to the worker for a safe working environment, in visual form. In addition to that, a future extension is possible to extend this system with an introduction of camera system, to monitor the safe-zones for human movement while working with the robot. Possible information to be visualized can be the position or state of the components, e.g. the robot, as well as the visualization of safety zones, current occupation, or visual indication of objects entering certain zones using blinking boxes or pigmented box. The information and error messages related to components, e.g. gripper information – product picked or rack information - load of boxes in racks can be visualized in detailed pictures corresponding to the different components. The provision of this information can be realized via standard OPC UA components, e.g. OPC UA servers for PLCs.

Implemented OPC UA related components in this scenario and resources used to this end were:

- Visualization client ProVis.Visu® and corresponding pictures for the scenario connected to the OPC UA server with integrated OPC UA Client written in C++ based on Unified Automation C++ SDK running on Windows at least 7 / Server 2008 R2

In this application scenario, the following OPC UA building block was used: Common info view

3.3. Scenario 3 – Management information board (monitoring and control)

This scenario demonstrates once again the ability of OPC UA to be used in safety critical systems even if OPC UA in its current state cannot be used under hard real-time constraints. An OPC UA solution was developed to monitor a demonstrator that facilitates human robot collaboration. This demonstrator consists of several modules where robots share working space with a human operator. For several tasks, it was necessary that human operator is informed about certain information needed for a safe working environment.

An OPC UA server, which runs on a Raspberry Pi, was designed that receives this information using several web-services from different modules. These modules are either connected through PLCs or other Raspberry Pi modules. OPC UA hides the specific web service based communication of the modules and standardizes this proprietary communication path. To this end, a special status web service was created for communication between modules and OPC UA server to avoid collision and affection in the normal operating mode of the production cell. This special web-service will collect all the data needed using other web-services, and then it will provide data to OPC UA server in JSON format (string). Each module has its own web service, and return strings for each module are different. The OPC UA solution was developed in such a way that future extension of the demonstrator is possible without modifying the implemented existing solution. An OPC UA client was designed that presents this information in visual form, as a graphical representation makes it safer for human operators by reducing the comprehending time.

Implemented OPC UA related components in this scenario and resources used to this end were:

- OPC UA Server written in C++ based on Unified Automation C++ based UA SDK for Linux (cross compiled for ARM architecture) and PicoJSON (C++ JSON parser) running on Raspbian (any version)

- Visualization client ProVis.Visu® and corresponding pictures for the scenario connected to the OPC UA server with integrated OPC UA Client written in C++ based on Unified Automation C++ SDK running on Windows at least 7 / Server 2008 R2

In this application scenario, the following OPC UA building block was used: Common info view

3.4. Scenario 4 – Orchestration of cyber-physical production systems (production cells)

This scenario demonstrates the ability of OPC UA to be used as generic interface for orchestration of components in a production cell. This can be used to achieve flexible software deployment in adaptive plants. The demonstrator consists of different hardware components, e.g. a robot and an optional camera in a robot cell which shall be orchestrated. Furthermore OPC UA can be used to encapsulate functionality in a neutral way, e.g. code analysis and other methods. OPC UA is ‘only’ a vehicle for describing status and results of components and software modules and providing access to the services of the software modules and components. If the properties of one component fit to the need of another component, e.g. a camera for a robot, the component is able to connect to the other component and to execute the analysed code during the production process. The state of the methods and their accessibility depend on the status of the system. The deploying method is e.g. only available if analysis was successful.

The orchestration OPC UA server was designed and developed to provide platform- and system-independent information via OPC UA. It receives input from a user or system which includes the description of the whole production scene to orchestrate. This input comes in form of a specific XML representation of RobotML (Robot Modelling Language) [14], which could be a future candidate for an OPC UA companion specification. The orchestration server holds this information and represents software and hardware as objects with executable methods in the OPC UA information model. The methods provide access to the services or functionality provided by the production hardware components (e.g. visual observation) or software modules (e.g. code analysis). It distributes the information by notifying all components, e.g. robot or camera, of the new scene description via OPC UA. If the scene description includes information about possible new functionality provided by some of the components, it triggers the execution of such functionality and stores the result for other components. If the result is matching with predefined requirements, it can trigger the setup of communication and coupling between the components of the production scene. To ensure the possibility for future extension, the OPC UA server was developed based on an open source OPC UA toolkit (http://open62541.org/) and the Lua scripting framework.

Additionally, a visual dashboard (overview of scene) exists to visualize the content of the OPC UA servers to the users (e.g. the operator). It is necessary to represent results of network orchestration or code analysis (e.g. release for deployment based on analysis of parallelization). Therefore, an OPC UA client was designed that enables to present this information in visual form. Other standard OPC UA clients on mobile platforms, e.g. tablets, can be used as well to provide access to the information via web or desktop.

Implemented OPC UA related components in this scenario and resources used to this end were:

- OPC UA Server and Client written in C and LUA scripts based on open62541 SDK and LUA scripting framework running on Linux x86_64 (Ubuntu ViVid 15.04)

- Visualization client ProVis.Visu® and corresponding pictures for the scenario connected to the OPC UA server with integrated OPC UA Client written in C++ based on Unified Automation C++ SDK running on Windows at least 7 / Server 2008 R2

In this application scenario, the following building blocks are used:

- Common info view
- Aggregating OPC UA server functionality
- AutomationML or other description model (in the present case: RobotML) as base/data source

3.5. Development resources for the listed scenarios

For the implementation of the four scenarios and their specific building blocks, different programming resources were used based on the given environment and requirements, which are depicted in Fig. 3 and Table 1. This shows the variety of OPC UA depending on hardware platform, operating system, programming language, license and cost model, etc.

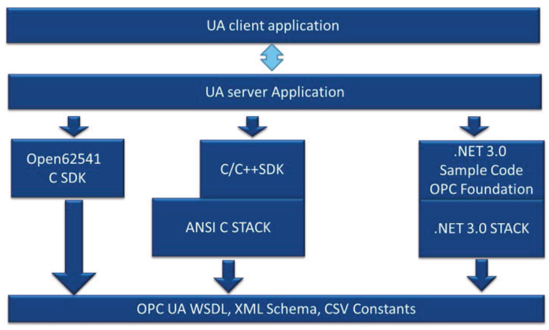

Fig. 3. Development resource options for OPC UA server/ client applications.

Table 1. Used development resources

Topic	Scenario 1	Scenario 2	Scenario 3	Scenario 4
OPC UA components	OPC UA server, OPC UA client	OPC UA client	OPC UA server, OPC UA client	OPC UA server, OPC UA client
Programming language	C++, C#	C++	C++	C, LUA
SDK or other development resources	Unified Automation C++ SDK, OPC Foundation Sample Code Version 1.02	Unified Automation C++ SDK	Unified Automation C++ based UA SDK for Windows and for Linux (cross compiled for ARM architecture)	Open62541 OPC UA SDK
Operating system	Windows at least 7 / Server 2008 R2	Windows at least 7 / Server 2008 R2	Linux Raspbian, Windows at least 7 / Server 2008 R2	Linux x86_64 (Ubuntu ViVid 15.04)
Specific Hardware	Windows tablet, Windows PCs/Mini-PC	Windows PC	Raspberry Pi, Windows PC	Linux PC

4. Summary and outlook

One challenge in using OPC UA is to select matching OPC UA software components and to specify a well-defined architecture for each of the use cases.

OPC UA can be used as data and information hub for dedicated purposes as shown in Scenario 1, e.g. for a quality defect tracking system. In this scenario, the persistence of the information was ensured by storing it into a database connected to the OPC UA server. The system based on OPC UA is much more flexible compared to a simple database application. Furthermore, it makes the visualization of different parts (views) of the information for different users (worker, manager) and applications (data collection, data analysis) possible.

The most likely OPC UA application scenario is the monitoring and control scenario. OPC UA is used to connect from production process, e.g. the PLC, to a visualization system for monitoring the production process status and other corresponding information. This was shown in scenario 2 and 3. Gathering of information from the production process can be realized directly via OPC UA if OPC UA servers are available on this level or via proxy functionality encapsulating e.g. specific web services via OPC UA.

The most "futuristic" scenario which shows the strength and flexibility of OPC UA was the last one (scenario 4). Here different platforms, different programming languages, different SDKs, different operating systems, and components from different vendors, etc. come together.

The upcoming specification for the usage of OPC under hard real-time requirements based on TSN and the Pub-Sub mechanism will for sure enlarge the possibilities of OPC UA und simplify many of the current applications in industry and research.

In future, the connection of OPC to other platforms and standards will be much more important. One example can be given related to the IoT domain: Due to the rapidly growing number of devices which produce a growing amount of data, an open-source middleware platform for IoT devices (OpenIoT) has been created [15]. The OpenIoT platform supports a broad range of internet-connected objects and provides automated configuration of filtering/fusion/reasoning mechanisms. OpenIoT tries to realize a fusion of Cloud Computing and IoT in the so called Sensor Clouds. It supports cloud paradigms for internet-connected objects and enables the user to configure, deploy, and use IoT based services. OpenIoT is based on three layers: the Global Sensor network (GSN) as an interface to the physical entities including X-GSN [16] the for semantic annotation of sensor data, the Linked Sensor Middleware (LSM) as a semantic middleware based upon RDF and SPARQL as well as the application layer with service utilities and application design and operational support tools. For the purpose of inter-company signal communication it was investigated how to integrate OpenIoT with the production-related standard OPC UA. Two or more spatially separated production sites (see Fig. 4) can be connected to each other via the OpenIoT platform.

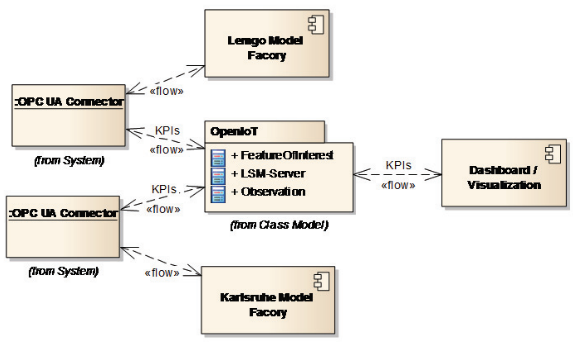

Fig. 4. Inter-factory communication.

An extension of the communication network (more than two communicating partners) is possible. A first idea for the use of the OpenIoT platform in industrial production results in replacing the X-GSN software in the physical layer by a software package that is tailored to OPC-UA. Furthermore, the ontology used in the virtualized layer of OpenIoT (currently SSNO) shall be extended, accordingly.

Acknowledgements

Parts of this contribution summarize the results of the joint research and development project "EIT Digital - High Impact Initiative CPS - INDUSTRIE 4.0 Powering Europe" which was finished in 2015 together with partners from Siemens, CEFRIEL, CEA, CRF, Aalto University, and DFKI. The authors want to thank all members of this project for their contributions. The authors are responsible for the content of this publication.

References

[1] Miriam Schleipen, Arndt Lüder, Olaf Sauer, Holger Flatt, Jürgen Jasperneite: Requirements and concept for Plug-and-Work - Flexibility in the context of Industry 4.0. at. Band 63, Heft 10, Seiten 801–820, ISSN (Online) 2196-677X, ISSN (Print) 0178-2312, DOI: 10.1515/auto-2015-0015, October 2015

[2] OPC UA, IEC 62541, standard series.

[3] VDI GMA 7.21 "Industrie 4.0" – Sub group "Terms" @ ONLINE, www.iosb.fraunhofer.de/?BegriffeI40

[4] Usländer, T. (Ed.): Industrie 4.0 - Auf dem Weg zu einem Referenzmodell. VDI Statusreport. VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, Düsseldorf, April 2014, http://www.vdi.de/industrie40, 2014.

[5] VDI, ZVEI: Status Report: Reference Architecture Model Industrie 4.0 (RAMI4.0), July 2015.

[6] Givehchi, Omid, and Jürgen Jasperneite. "Industrial Automation Services as part of the Cloud: First Experiences. Jahreskolloquium Kommunikation in der Automation - KommA, Magdeburg, 2013"

[7] Imtiaz, Jahanzaib, and Jürgen Jasperneite. "Scalability of OPC-UA down to the chip level enables "Internet of Things"." 11th IEEE International Conference on Industrial Informatics (INDIN) 2013. IEEE, 2013.

[8] Miriam Schleipen, Olaf Sauer, Jing Wang: Semantic integration by means of a graphical OPC Unified Architecture (OPC UA) information model designer for Manufacturing Execution Systems. In: Sihn/Kuhlang (Ed.), Sustainable Production and Logistics in Global Networks. 43rd CIRP International Conference on Manufacturing Systems, 26-28 May 2010, Vienna. ISBN 978-3-7083-0686-5, Neuer Wissenschaftlicher Verlag GmbH NfG KG, pp. 633-640, 2010.

[9] Miriam Schleipen: Adaptivität und semantische Interoperabilität für Manufacturing Execution Systeme (MES)", Karlsruher Schriften zur Anthropomatik, ISBN: 978-3-86644-955-8, 2012.

[10] Robert Henssen, Miriam Schleipen: Interoperability between OPC-UA and AutomationML.Disruptive Innovation in Manufacturing Engineering towards the 4th Industrial Revolution, Proceedings of the 8th International CIRP Conference on Digital Enterprise Technology - DET 2014, Fraunhofer Verlag, ISBN 978-3-8396-0697-1, 2014. Full-text: Procedia CIRP, Volume 25, Pages 297–304, 2014.

[11] Julius Pfrommer, Miriam Schleipen, Jürgen Beyerer: PPRS: Production skills and their relation to product, process, and resource. In: Proceedings of the 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA), September 2013.

[12] Pfrommer, J., Stogl, D., Aleksandrov, K., Escaida Navarro, S., Hein, B., & Beyerer, J. (2015). Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems. at-Automatisierungstechnik, 63(10), 790-800.

[13] Dr. Dirk Schulz, Roland Braun, Dr. Johannes Schmitt – OPC UA in practical applications. In: ATP Edition 2015-09, p26-35, Vulkan Verlag, ISSN 2190‑4111

[14] S. Dhouib, S. Kchir, S. Stinckwich, T. Ziadi, M. Ziane : "RobotML, a Domain-Specific Language to Design, Simulate and Deploy Robotic Applications", 3nd International Conference on Simulation, Modeling and Programming for Autonomous Robots, SIMPAR 2012, Lecture Notes in Computer Science, pp. 149-160, (Springer-Verlag) (2012)

[15] Open IoT platform @ ONLINE, https://github.com/OpenIotOrg/openiot

[16] Jean-Paul Calbimonte, Sofiane Sarni, Julien Eberle and Karl Aberer. "XGSN: An Open-source Semantic Sensing Middleware for the Web of Things". 6th International Workshop on the Foundations, Technologies and Applications of the Geospatial Web and 7th International Workshop on Semantic Sensor Networks co-located with 13th International Semantic Web Conference (ISWC 2014), pp. 51-66, 2014.