Enabling Plug&Play Cyber-Physical Systems Using Knowledge-Driven OPC UA Discovery

Václav Jirkovský, Petr Kadera
Czech Institute of Informatics, Robotics, and Cybernetics
Czech Technical University in Prague
Prague, Czech Republic
Email: {vaclav.jirkovsky, petr.kadera}@cvut.cz

Marek Obitko
Rockwell Automation R&D Center
Argentinska 1610/4
Prague, Czech Republic
Email: mobitko@ra.rockwell.com

***Abstract*—Industrial automation domain is changing rapidly in recent years. It is very important in the involvement of technologies such as Cyber-Physical Systems. They are based on interoperable devices. On the other hand, it is difficult to find a suitable format and mechanism for successful communication of devices. One of the important requirements is to find a specific device as a demanded counterpart for communication. In this paper, we will show that the integration of OPC UA standard and Semantic Web technologies provides an interesting solution.**

***Index Terms*—Cyber-Physical System, Ontology, Discovery, OPC UA, Industrial Internet of Things**

I. Introduction

The industrial automation domain has changed in recent years. The changes are labeled as the industrial revolutions. The last one is the 4th industrial revolution well-known as Industry 4.0. Industry 4.0 is based mainly on digitization and virtualization.

Furthermore, Internet of Things (IoT) [1] is one of the next cornerstones of Industry 4.0. IoT is one of the Cyber-Physical Systems (CPSs) variants with a typical purpose and architecture — a set of interoperable devices connected via a suitable way to control a given physical process or processes.

The important obstacle for full adoption of IoT as well as CPS(s) is communication among devices. Concerning complex analytic tasks, it is difficult to find the desired device using available standards. In this paper, we will show that the integration of CPS and Semantic Web Technologies (RDF[1] and OWL[2]) may provide the required functionality.

The paper is organized as follows: first, we provide a general overview of Cyber-Physical Systems and OPC UA standard with respect to OPC UA discovery. Then, we introduce the developed prototype named Semantic Big Data Historian together with Plug&Play concept. Next, we introduce an extension of OPC UA discovery functionality using Cyber-physical system Ontology for Component Integration. Finally, we describe details of the semantic matchmaking of devices.

This research has been supported by Rockwell Automation Laboratory for Distributed Intelligent Control (RA-DIC) and by institutional resources for research by the Czech Technical University in Prague, Czech Republic.

[1]https://www.w3.org/RDF/

[2]https://www.w3.org/OWL/

978-1-5386-4980-0/19/$31.00 ©2019 IEEE

II. State of the Art

This paper will introduce a semantic extension of OPC UA discovery for Cyber-Physical System. Cyber-Physical Systems and OPC UA are described in the following paragraphs.

A. Cyber-Physical Systems

Around 2006, the term "Cyber-Physical System" (CPS) was coined by Hellen Gill at the National Science Foundation (U.S.) to describe the integration of physical and computational processes [2].

In general, CPSs may be considered as the outcome of the long-standing evolution which was triggered by the creation of the first computer. Computers were initially invented to facilitate and perform a computation. The evolution from a tool for computation to a complex system integrating computing, communication, and control technologies, was enabled by many essential inventions which made possible purposes and utilization of the system wider.

The basic concept of CPS architecture consists of three main parts — a cyber part, a physical part, and a network:

- The cyber part represents a computing core where physical process information is transformed into a a model of a software system and corresponding rules establishing dependencies and relationships among software model entities together with control algorithms.
- The physical part represents a controlled object. This part involves physical processes and physical objects which are linked according to the given process.
- The network represents a communication medium between a cyber and a physical part.

B. OPC UA

OPC UA [3] provides means for platform independent, versatile, and robust client-server communication. It replaces the old OPC Classic protocol that was built upon Microsoft's DCOM (Distributed Component Object Model) technology with a platform independent approach. There are not only technological differences but also conceptual changes enabling smoother vertical and horizontal integration. It defines a means for scalable and secure industrial solutions.

A key feature of OPC UA is its capability to find available data sources. This feature is reffered as Discovery Service.

The discovery process is related either to a single host (Local Discovery) or to a network connecting multiple hosts (Global Discovery).

Discovery services are provided by Discovery Servers (DSs). DSs enable OPC UA servers to register. Once a client sends a "Finds Servers" request the Local Discovery Server (LDS) responds with a list of registered servers. Each server is described by a set of attributes including:

- Application Uri: ID of the server instance
- Application name: Human readable name for the server
- ProductUri: ID of the server product
- DiscoveryUrls: The available URLs of the server that allow calling GetEndpoints without requiring a secure connection

The client uses these pieces of information to select which UA servers are of its interest. OPC UA does not provide any standardization of this process. The approach proposed in this paper aims at this gap. It uses Application Uri for matching a specific data source represented by an OPC UA server and the ontology managing data within the Semantic Big Data Historian (SBDH). Thus, semantic search can be used to effectively look up a particular sensor and at the same time use standard OPC UA services to access the corresponding OPC UA server and exchange data.

III. Plug&Play Concept Realization for Semantic Big Data Historian

The Semantic Big Data Historian was proposed and prototype implemented due to increasing demands for facilitating flexible manufacturing. The core functionality, as well as the main advantage of the historian, is the employment of Semantic Web technologies (more precisely an ontology in OWL[3]) for explicit definition of knowledge. Thus, specific requirements for a historian architecture stem from a utilization of the ontology. Furthermore, the architecture is influenced by a historian target usage, i.e., gathering data and information from a shop floor and other involved systems as well as controlling a shop floor by appropriate feedback. Thus, the architecture has to be very flexible to process all required data and robust to provide a highly reliable solution.

Frameworks employed in SBDH have changed several times according to changing requirements and available software. The current implementation exploits Apache Spark[4] for coping with data streams and big data management and Apache Cassandra[5] for data storage. SBDH architecture consists of four layers and a concept of the overall system is to provide a modular solution which may be adapted according to given needs and requirements for software. The following listing provides the description of the four SBDH architecture layers:

- Data acquisition and control layer — this layer is responsible for acquisition of data from relevant sources (e.g., sensors, users via a user interface, and any relevant software from higher levels such as MES/ERP) and providing a feedback to control a given process (e.g., controlling an actuator, call relevant services of 3rd party system, etc.). The preferred way for communication is using previously introduced OPC UA. Consuming of data streams is solved using Spark Streaming Custom Receivers[6].
- Transformation layer — a transformation to a form of RDF triples according to Cyber-physical system Ontology for Component Integration (COCI) [4]. A very important responsibility of the transformation layer is to solve semantic heterogeneity and repair damaged data if possible.
- Data storage layer — transformed data in the form of RDF triples are stored in a triple-store in this layer. The storage respects nature of prevalent part of data, i.e., measurements from sensors. In general, two different file models are used in SBDH to provide more homogeneous data distribution across files in Cassandra — "vertical partitioning" model for data which are not time series (triples are partitioned according to a predicate of the triple) and "hybrid SBDH model" for storage of time series (triples are partitioned according to a triple predicate and a given sensor). More detailed description is available in [5]. It is obvious this layer is not only responsible for simple data storage but is also responsible for conducting transformations of triples to a corresponding file model.
- Analytic layer — the last layer provides a means how to access data stored in the triple-store with the help of SPARQL[7] and how to implement analytic tasks. Apache Spark MLlib[8] (library with implemented distributed machine learning algorithms) for a solution of analytic tasks.

The overall architecture is shown in the Fig. 1 and more details may be found in [6].

A. Plug&Play Concept

An interesting feature of SBDH resulting from the utilization of Semantic Web technologies for knowledge model is the feature named "Plug&Play concept". It benefits from the explicit specification of knowledge. Every device connecting to SBDH has a semantic annotation stored in a string property of OPC UA information model. This semantic annotation consists of triples identifying a device against COCI concepts. Furthermore, if a user wants to connect a previously unknown device then the semantic annotation property may provide definition of the device (including not only type of device but also other related properties and capabilities such as sampling rate, range, etc.) to extend ontology definition if more general concepts from COCI (e.g., the concept named "sensing device") are respected.

Furthermore, Plug&Play devices may be directly employed in analytic tasks or applications connected to SBDH without any additional configuration or changing algorithms. The direct employment of devices is achievable in the cases when

[3]Web Ontology Language - https://www.w3.org/OWL/
[4]https://spark.apache.org/
[5]http://cassandra.apache.org/
[6]https://spark.apache.org/docs/2.2.0/streaming-custom-receivers.html
[7]https://www.w3.org/TR/rdf-sparql-query/
[8]https://spark.apache.org/mllib/

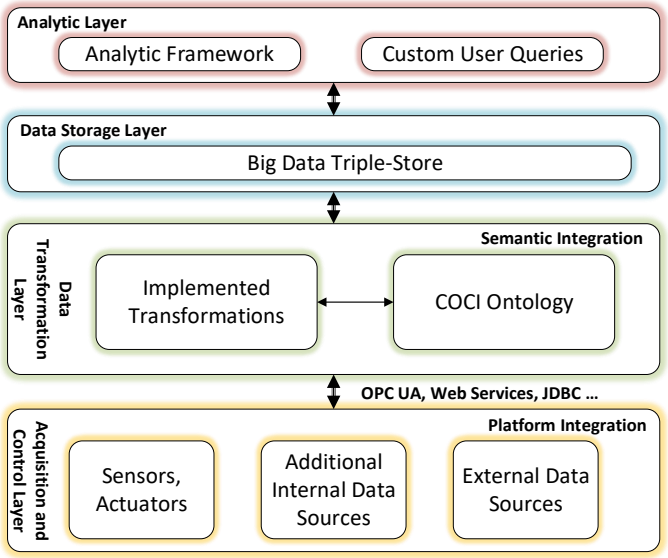

Fig. 1. Semantic Big Data Historian architecture

they are referenced "relatively" instead of "absolutely", e.g., humidity sensors from a particular room_A and their measurements may be acquired by querying SBDH using following query — "get all humidity sensors from room_A and get their measurements"; instead of their direct referencing using their IP address, etc.

The basic idea of Plug&Play was roughly introduced in preceding paragraphs and more detailed description of the Plug&Play concept may be found in [7].

Obviously, there are several aspects which cannot be generally solved. One of these aspects is security. Security solution (authorization and authentication) for a connection of the device to SBDH and subsequent communication is application sensitive, and every application, as well as every company, may have specific requirements for the security. In our solution, certificates are used during device connection to SBDH.

The other important requirement on SBDH is ensuring easy discovery functionality of connected devices capabilities. This topic is discussed in following paragraphs.

IV. OPC UA Discovery in Semantic Big Data Historian

OPC UA Discovery is one of the enablers for the introduced Plug&Play concept. It provides and manages a device registration as well as establishing a connection to Apache Spark. On the other hand, there are several limitations in OPC UA discovery capabilities according to OPC UA standard, and they are discussed in following paragraphs.

As previously mentioned, data streams are received using the OPC UA client (based on Eclipse Milo[9] OPC UA standard implementation) which is implemented as Spark Streaming Custom Receiver. If SBDH only reads data streams provided by devices and any other functionality is expected from SBDH, then everything needed may be solved and managed by standard OPC UA global discovery server, OPC UA client, and Apache Spark. However, communication between devices and SBDH is bidirectional (SBDH requires to control devices — e.g., actuators) and furthermore the feedback may be a target at various relevant devices. In other words, the feedback is not controlled by an OPC UA client (a receiver of SBDH), but it is controlled by the analytic layer of SBDH. Therefore, there should be a way to find a corresponding device. For example, a control algorithm decides to close valves of a heating system in specific rooms of a building based on data from an outside temperature sensor and a light sensor.

The high level SBDH discovery architecture is illustrated in Fig. 2. Unfortunately, the current standard does not provide a way to find a specific OPC UA server based on its capabilities which are explicitly specified in machine readable format.

A. Knowledge-Driven OPC UA Discovery

Because of aforementioned reasons, an integration of OPC UA discovery and COCI ontology seems to be able to provide demanded functionality for SBDH. Every connected device, such as sensor, actuator, or even more complex devices, have a corresponding concept in COCI, which represents a given type of the device, and a corresponding individual, which represents a particular realization of the concept — a given particular device. The *Concept URI* from COCI is identical to *Product URI* of the discovery *FindServer* method response and *Individual URI* is identical to *Application URI*.

The simplified mechanism of the knowledge-driven OPC UA discovery is illustrated in Fig. 3. The workflow from an actuator point of view is summarized in following steps:

1) Device registration — first of all, the actuator has to be registered using OPC UA discovery server. The actuator shall send common information needed for the discovery (where Product URI is a corresponding Concept URI and Application URI is a URI of a given individual) accompanied by the semantic metadata. These metadata includes a reference to a OPC UA method which is related to a possible action of the actuator according to COCI ontology.
2) Decision from the analytic layer — analytic tasks performed by the analytic layer may result in a requirement to control particular devices. The analytical layer has all needed information about individuals (i.e., individual URIs, correspondent concept URIs, and possible actions which may be performed by devices). The OPC UA client is called from this layer to find relevant OPC UA servers of devices using OPC UA discovery.
3) Find servers request — OPC UA client calls OPC UA discovery server. There are two parameters in *FindServer* request. None of them is suitable for filtering servers for a response. Only *Server URL* may be exploited but we prefer "relative" references instead of "absolute" references due to possible changes in URLs of device.
4) Filtering find server response — if the OPC UA client receives a list of OPC UA servers then it is filtered using

[9]https://projects.eclipse.org/projects/iot.milo

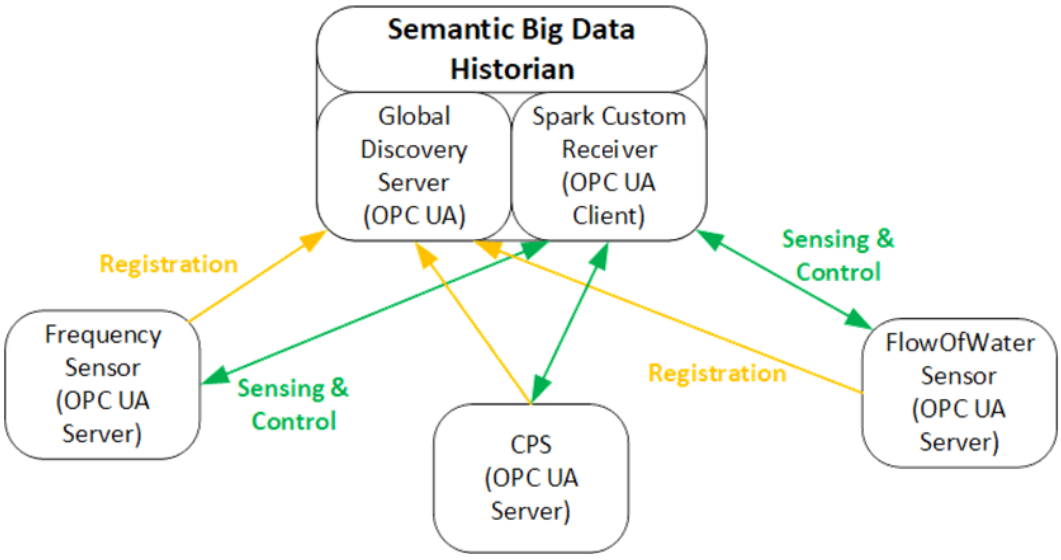

Fig. 2. OPC UA discovery in Semantic Big Data Historian

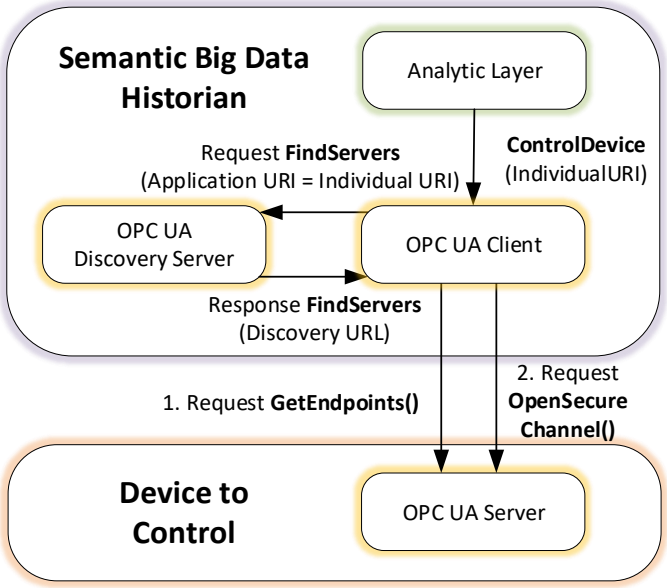

Fig. 3. Semantic Big Data Historian — Discovery workflow

specific URIs representing individuals (devices) or using a URI of a given concept to obtain a specific type of devices.

5) Connection to devices — the OPC UA client has to establish connections to devices. First, it calls *GetEndpoints* request to obtain a list of available endpoints. Next, the client chooses an appropriate endpoint. Finally, the client calls *OpenSecureChannel* request. If everything was processed without any problem, then the connection is established.

The described extension of OPC UA discovery, which provides possibilities to find devices according to their capabilities, may be understood as semantic matchmaking which is shifted from the discovery server to SBDH analytic layer, COCI ontology, and a reasoner. The semantic matchmaking is described in the following paragraphs.

B. Semantic Matchmaking

The fact that the Plug&Play CPS devices (sensors, actuators, or their integrations) are described in OWL allows semantic matchmaking, for searching appropriate connected sensors. This can be used for any querying within SBDH, such as user queries, looking for appropriate sensors or devices for a job at hand, including possible substituion or replacement, for diagnostics purposes, etc.

In general [8], when the semantic description of a sensor is D and the query request is R, then the following levels can be recognized:

- **Exact.** When $D \equiv R$, i.e., when D and R are equivalent concepts, then the sensors are exactly the same. This is the best situation, however, such match may not be found for a particular request and available sensors.
- **PlugIn.** When $R \sqsubset D$, i.e., when R is a subconcept of D, then the match level is PlugIn. An example is that the request R is a humidity sensor with sampling frequency 1 hour and the sensor D is a humidity sensor sampling frequency 1 second. In this case the D certainly satisfies all the R requirements, it provides even more. That is why the D can be "plugged-in" to request R, without any side-effects. This is usually also a good situation — something usable for the query was found, but unlike Exact match, it can be at additional cost that may be not necessary.
- **Subsume.** When $D \sqsubset R$, i.e., when R is a superconcept of D, then the match level is Subsume. An example is that the described sensor D is a humidity sensor with operating temperature range $0 - 40\,^{\circ}\mathrm{C}$ and the request R a humidity sensor with operating temperature range $0 -$

$100\,^{\circ}\mathrm{C}$. The matched sensor D is satisfying the request R only partially — there is some match, D can be used to some degree, but not all the requested features are available.

- **Intersection.** When $\neg(D \sqcap R \sqsubset \bot)$, i.e., when the intersection between R and D is not empty, then the match level is Intersestion. This means that there is some overlap between the requested and the matched sensor, such as in the case when the request R is humidity sensor able to work in explosion hazard area and having operating temperature range $0 - 100\,^{\circ}\mathrm{C}$ and sampling frequency 1 hour, while the sensor D is a humidity sensor with sampling frequency 1 second and operating temperature range $-20 - 20\,^{\circ}\mathrm{C}$. The intersection is a humidity sensor with operating temperature range $0 - 20\,^{\circ}\mathrm{C}$ and sampling frequency 1 hour — i.e., the request R is not only not fully satisfied, but in addition there is some additional functionality that is not needed and may bring unnecessary costs or side effects. Nevertheless, the intersection is not empty.
- **Disjoint.** When $D \sqcap R \sqsubset \bot$, i.e., when there is no intersection, the matchmaking failed. For example, when the sensor D is a temperature sensor and the request R is a drilling machine, then there is no overlap and there is no practical way of satisfying the request R.

Note that the levels are sorted according to their usefulness — the Exact match is the best match, PlugIn is the second best match, etc., till Disjoint which is the worst situation.

V. Conclusions

In this paper, we have introduced the prototype solution for knowledge-driven OPC UA discovery which is a part of Semantic Big Data Historian. The extension of OPC UA discovery is based on the integration of Semantic Web technologies with OPC UA and specific SBDH architecture.

A common problem is how to find a device based on its capabilities. In the previous sections, a solution is presented showing that ontologies may solve this problem. This OPC UA discovery extension may be perceived as the semantic matchmaking with the help of OWL and a DL reasoner.

This extension may have important benefits for facilitating flexible manufacturing. The semantic matchmaking may find an alternative device in cases when a precisely required device is unavailable.

Furthermore, we hope that the part of the presented problem (a shared conceptualization based on specific devices standardization which is in our case represented by COCI ontology) will be solved in the near future by OPC Foundation. Such a shared conceptualization provided by some respected entity such as OPC Foundation should guarantee high interoperability of devices originated from various manufacturers.

In future work, we would like to focus on more complex systems — represented by integration of various sensing devices and various actuators. Furthermore, the problem of how to verify the suitability of an alternative device automatically should be solved, e.g., the presented use-case where a sensor with higher sampling frequency is an alternative sensor for a sensor with a lower frequency cannot be suitable for every application.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] E. A. Lee and S. A. Seshia, *Introduction to embedded systems: A cyber-physical systems approach*. MIT Press, 2016.

[3] (2017) OPC Unified Architecture Specification. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture

[4] V. Jirkovský, "Semantic integration in the context of cyber-physical systems," Ph.D. dissertation, Czech Technical University in Prague, 2017.

[5] V. Jirkovský and M. Obitko, "Enabling semantics within industry 4.0," in *Industrial Applications of Holonic and Multi-Agent Systems*, V. Mařík, W. Wahlster, T. Strasser, and P. Kadera, Eds. Cham: Springer International Publishing, 2017, pp. 39–52.

[6] V. Jirkovský, M. Obitko, and V. Mařík, "Understanding data heterogeneity in the context of cyber-physical systems integration," *IEEE Transactions on Industrial Informatics*, 2016.

[7] V. Jirkovský, M. Obitko, P. Kadera, and V. Mařík, "Towards plug play cyber-physical system components," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2018.

[8] L. Li and I. Horrocks, "A software framework for matchmaking based on semantic web technology," *Int. J. Electronic Commerce*, vol. 8, no. 4, pp. 39–60, 2004. [Online]. Available: https://doi.org/10.1080/10864415.2004.11044307