computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

INTEGRATION OF BACNET OPC UA-DEVICES USING A JAVA OPC UA SDK SERVER WITH BACNET OPEN SOURCE LIBRARY IMPLEMENTATION

Naksit Anantalapochai [1)], Axel Sikora [1)], David Eberlein [2)], Dominique-Stephan Kunz [2)]

[1)] Hochschule Offenburg, Badstrasse 24, D77652 Offenburg, Germany
nanantal@stud.hs-offenburg.de, axel.sikora@hs-offenburg.de
[2)] Fr. Sauter AG, Im Surinam 55, CH4058 Basel, Switzerland
{david.eberlein; dominique.kunz}@ch.sauter-bc.com

Abstract: *The variety of technologies used in modern Building Automation Systems (BAS) calls for methods to support interoperability of the devices from different technologies and vendors. OLE for Process Control Unified Architecture (=OPC UA) provides the possibility to enable secure interoperability of devices with platform independence and efficient information model features. However, OPC has not found broad space in the world of building automation, yet.*

In this paper, results and experiences from a project are presented, where BACnet devices are mapped to OPC UA standard models. The values and controls are presented by the OPC UA server running on an embedded device. In this paper, we map the BACnet information models into the corresponding OPC UA information models. The actual data (in OPC UA form) of the BACnet devices can be accessed by connecting an OPC UA Clients to the OPC UA Server. This objective was be pursued by using as many available open-source projects as possible.

Keywords: *BACnet, OPC Unified Architecture, Building Automation Systems, OPC UA Server*

1. INTRODUCTION

One of the major challenges for developers and integrators of modern Building Automation Systems (BAS) is the integration of different technologies and devices from numerous vendors. Interoperability of these devices is required to integrate all devices and all information into the same interworking model and into one server for improved controllability and observability. This allows the reduction of: cost, installation efforts, system complexity and increases reliability. OLE (Object Linked and Embedded) for Process Control Unified Architecture (=OPC UA) can be used to present a generic view to the monitoring clients that need access to the entire BAS. Using a Java based SDK for the OPC UA Server is an additional promising stepping stone for the platform independence.

The implementation of an OPC information model for the BACnet types and objects is generally the most significant task in this process, and also of this project. This paper describes one of the basic approaches of BACnet-to-OPC UA integration and a sample implementation. Furthermore, as the server will be running constantly, energy consumption of the control system must also be taken into account. Therefore, the presented project uses a programmable embedded device for the OPC server, which connects to the BACnet devices and the OPC UA client for monitoring in the same local area network.

This paper is structured as follows: After the description of some parallel activities in ch. 2, a brief overview on the OPC UA and the BACnet standard is given in chapters 3 and 4. The following ch. 5 describes the method to setup and prepare the devices and the working environment. Ch. 6 elaborates the use of the “BACnet4J” Java library to access BACnet services, before ch. 7 describes the setup of Java OPC UA Server and Client, as well as the monitoring of the BACnet device values. Finally, ch. 8 reports on the implementation and its results.

2. STATE OF THE ART

To the best knowledge of the authors, the most up-to-date procedure to create an OPC information model for the BACnet data model was presented in [1] and [2], which covers the most important BACnet information (object and property) types, i.e. the mandatory types.

With this initial description it gives an inspiration to research for an optimized way for industrial

operation and commissioning. As stated in the introduction section a programmable embedded device is used for the operation to fulfill these requirements. Due to the restrictions in memory and processing power, this project implements the integration in a simplified way, using selected concepts and ideas from [1] and [2].

3. OPC UA

3.1 OVERVIEW

Object Linking and Embedding (OLE) for Process Control (=OPC) was developed in 1996 by the *OPC Foundation*, an association of worldwide industrial automation suppliers working in cooperation with Microsoft. Now, OPC is an open standard specification that describes the communication of real-time plant data between control devices from different manufacturers.

The origin of OPC is based on OLE, Component Object Model (COM) and Distributed Component Object Model (DCOM) technologies developed by Microsoft for Windows operating systems. The very first standard from the *OPC Foundation* was for Data Access (OPC DA). Soon after OPC DA was launched, it was realized that communicating other types of data could benefit from standardization. Thus, standards for Alarm and Events (OPC AE), Historical Data Access (OPC HDA), Data Exchange (OPC DX) and Batch Data were published later on.

The fact that legacy OPC standards are based on Microsoft's COM/DCOM paradigm later turned out to be a limitation to several operations especially for interoperability. Besides, the insufficiency of the object oriented concept it was impractical to model complex data structures.

Because of these restrictions, a new standard was needed. Eventually, OPC Unified Architecture (OPC UA) was released in 2009, it extends the OPC communication protocol and enables data acquisition, information modeling, reliable and secure communication between the plant floor and the enterprise resource planning level. It was intended to be a full replacement of the classic OPC specifications. The key features and benefits of OPC UA are:

- platform independence to run on any operating system including embedded devices
- single set of services to expose OPC data models (DA, AE, HDA, Batch, DX, etc.)
- a reliable, secure and efficient way to transport higher level structured data
- a broader scope of connectivity
- extensible for future applications and modifications
- backward compatibility

The overall OPC UA specifications consist of 13 specification parts shown in Figure 1 [3].

It is important to understand how to model the address spaces and information, which is described in Part 3 "Address Space Model" and Part 5 "Information Model".

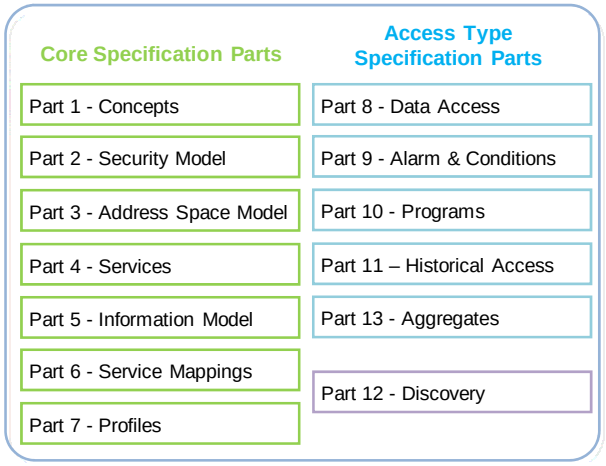


Fig. 1 – Parts of the OPC UA Specifications [3]

These two specifications are the key documents for the design and development of an OPC UA server to integrate with other standards and protocols. Before describing the information and address space modeling in OPC UA, some basic infrastructures (described in specifications Part 1 "Concept") should be introduced first.

The specification of common OPC UA Client and Server consist of two parts, the UA application and the communication Stack (cf. Fig. 2).

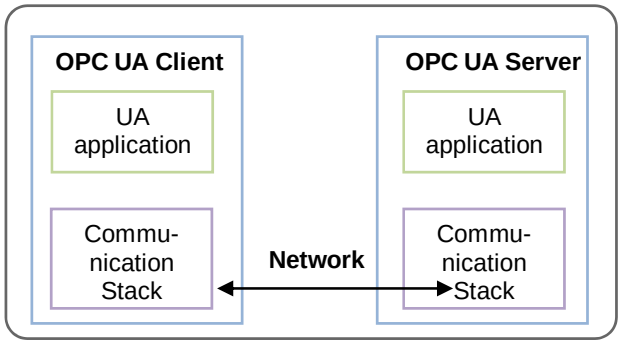


Fig. 2 – Common OPC UA Client and Server [3]

In the OPC UA Server, the application part contains the OPC UA address space representing the UA complex information model as a node. Figure 3 illustrates the complete architecture of an OPC UA Server.

3.2 INFORMATION AND ADDRESS SPACE MODELING

OPC UA represents information in the form of node hierarchies. It is necessary to understand the

basic concept of the OPC UA nodes to model data information. To enable interoperability between devices from different vendors, a uniform representation of data is required, which is called information model in OPC UA. The devices from any vendor can use or even inherit (with regards to the concept of object orientation) the model for their own usage. However before having an information model, first a place must be instantiated to store this information model.

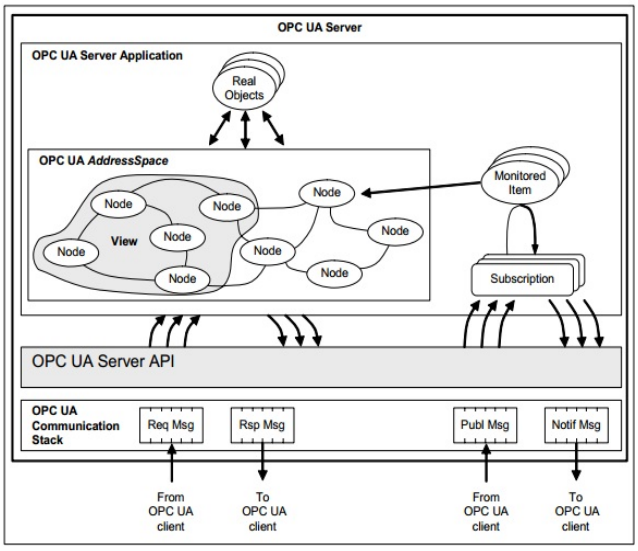


Fig. 3 – OPC UA Server Architecture [3]

Every information model is represented in the address space as a set of nodes which are interconnected with references in a hierarchical form (branches of a tree), where the highest node is called the 'Root' node. Figure 4 describes the node model and the details of the address space node model.

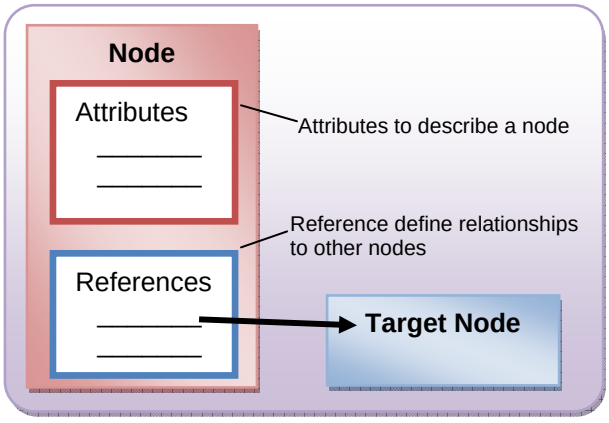


Fig. 4 – OPC UA Address Space Node Model [3]

When a node is instantiated in the address space, it is defined by a 'Node Class'. These node classes are referred to collectively as the metadata of the address space. The node classes can be classified to types and instances as stated in Table 1.

Table 1. OPC UA built-in definition node classes [3]

OPC UA built-in type definition node classes	
Data Type Node Class	Describes the structure of the Value Attribute of Variables and their Variable Types
Variable Type Node Class	Provides type definitions for Variables
Object Type Node Class	Provides type definition for Objects
Reference Type Node Class	Specifies References
OPC UA built-in instance definition node classes	
Object Node Class	Represents systems, system components, real-world objects and software objects. Defined by Object Type
Variable Node Class	Represent Values which may be simple or complex. Defined by Variable Type
Method Node Class	Defines callable functions. Invoked using Call Service defined in Part 4 of OPC UA Specification
View Node Class	Defines a subset of the Nodes in the Address Space in order to limit the node visibility for some specific purpose. All Nodes contained in the View shall be accessible from the View node when browsing

The OPC UA built-in '*DataTypes*' are all inherited from the highest parent node of *BaseDataType* and can be used directly for this work i.e. *Boolean*, *Integer(Number)* and *Float(Number)*. The same holds true for '*VariableTypes*' and '*ObjectTypes*', they are inherited from *BaseVariableType* and *BaseObjectTypes*, respectively. For the 'ReferenceTypes', there are two highest parent nodes: *HierarchicalReferences* and *NonHierarchicalReferences*.

4. BACNET

4.1 OVERVIEW

In June of 1987, BACnet was introduced as a communication protocol for building automation and control networks by the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE, http://www.ashrae.org). BACnet was designed to allow communication between building automation and control systems i.e. heating system control, air control, lighting control, access control, fire detection and any other control systems related to building automation. The BACnet protocol provides services for programmable/electronic building automation devices to exchange data.

The BACnet protocol has been frequently updated. The last update was made in October 2012. Therefore, it is recommended to regularly check for updates from the ASHRAE SSPC 135 BACnet official website [4].

The BACnet layered architecture is based on a reduced ISO-OSI Reference Model [6] shown in Figure 5 [5].

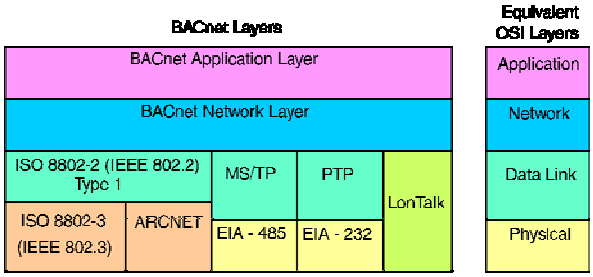


Fig. 5 – BACnet OSI-based Reduced Architecture [5]

The Application Layer is the most relevant layer to this project. This layer defines the BACnet information model and the BACnet services which will be used to poll the data from the BACnet device and then to process it into its information model before integrating it into an application (for instance OPC UA server).

4.2 BACNET: OBJECTS AND SERVICES IN APPLICATION LAYER

The application Layer contains information data consisting of objects and their properties as well as services used to exchange information through the layers below.

Each BACnet object has a type to distinguish the kind of the object. Up to date, there are already 50 object types defined within the standard. Within this project, five different object types are implemented, i.e. Analog Input, Analog Output, Binary Input, Binary Output and Device. An example of the Analog Input object, is shown in Fig. 6.

Object

Object_Type:	ANALOG_INPUT
Object_Name:	Temperature_1
Object_Identifier:	ANALOG_INPUT1
Present_Value:	23.1
Status_Flags:	In_Alarm
Event_State:	Normal, Out-of-Service
Units:	Degrees-Celsius
Out_Of_Service:	False

Fig. 6 – Example of a BACnet Analog Input Object

This example of a BACnet Analog Input object shows the object characterized by its mandatory properties. Different object types will have different mandatory properties together with the possibility to add optional properties.

From [7], in order to exchange information, BACnet services are classified into four different categories: Alarm&Event, File Access, Object Access and Remote Device Management. With regards to the project, only services from Object Access and Remote Device Management services are utilized.

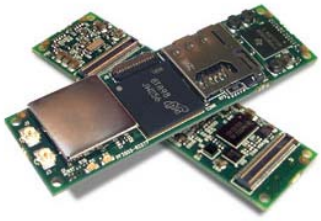
Remote Device Management services are used to discover BACnet devices in the network. Generally, a “Who-Is” request is broadcasted on the network and then waited for an “I-Am” reply from the BACnet devices, if they are present in the network.

Object Access services are used to access the objects of the device and to read or write the values of their properties. Examples of Object Access services are the ‘ReadProperty’, ‘WriteProperty’, ‘ReadPropertyMultiple’ and ‘WritePropertyMultiple’ services.

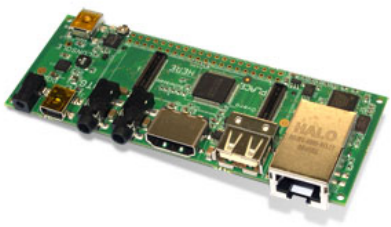
5. DEVICES AND SYSTEMS PREPARATION

5.1 OPC UA SERVER

As stated in the introduction section, an embedded device is used for the OPC server. This project takes an Overo Air Computer-on-Module (COM) with Tobi expansion board from Gumstix into operation. A brief overview of the device and its expansion can be found in the Table 2. The specification sheet of the devices can be accessed from the Gumstix official website in the product section for more details [8] [9].



a) Gumstix Overo Air COM



b) Gumstix Tobi Expansion Board

Fig. 7 – Devices for running OPC UA Server [8] [9]

Table 2. Device Overview

Gumstix Overo Air Computer-on-Module	
Architecture	ARM Cortex-A8
Processor	Texas Instruments OMAP3503 Applications Processor
Address	10, Times New Roman, Regular
CPU Speed	600 MHz
RAM	512 MB
NAND Flash	512 MB
Performance	up to 1,200 Dhrystone MIPS
Power Mgmt	Texas Instruments TPS65950
Gumstix Tobi Expansion Board	
Network Socket	10/100BaseT Ethernet
DVI-D (HDMI)	HDMI
USB Slots	USB Host USB On-the-Go (OTG) USB Console
Power	3.5V – 5V

The Gumstix Overo Air COM is mounted on the Gumstix Tobi Expansion Board for the primary purpose to connect it to the Ethernet network and to the power supply. The second purpose is to be able to locally control and monitor the functionalities of the OPC UA Server (e.g. mouse/keyboard connection, display on monitor screen).

Ubuntu 10.04 Lucid Desktop-Lite is installed as operating system due to its capability to have additional programs and easy configuration in the future. Nevertheless, other operating systems supported by Gumstix, like Angstrom, Android, Debian, or Windows CE would also be feasible. Since the OPC UA server is Java based, a Java Runtime Environment (JRE) is required in the system.

5.2 OPC UA CLIENT

Although the OPC UA Client is not the main focus of the project, it is indispensable for monitoring the result of the approach and in the real operation.

A simple PC or mobile device which can connect to the OPC Server as a Client in the network is the only requirement. It is also possible to have an OPC UA Client in the embedded device which runs the OPC UA Server itself and to communicate over local loopback.

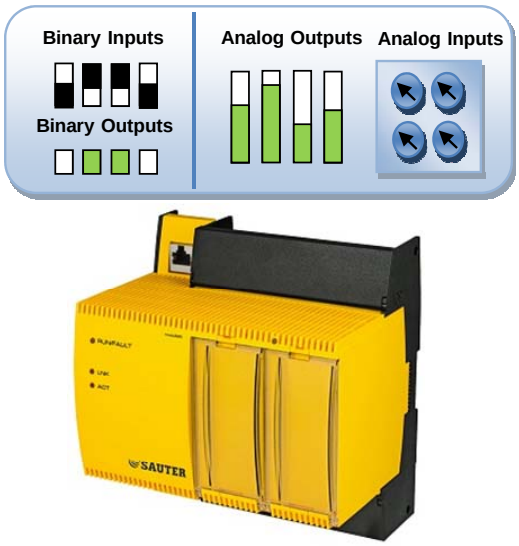


Fig. 8 – Basic configuration of the BACnet Automation Station SAUTER EY-modulo 525 [10]

5.3 BACNET DEVICE

At least one BACnet device is required in the network in order to provide the OPC UA server with information via the BACnet protocol. The project focuses on the 'Present_Values' information of Analog Input/Analog Output/Binary Input/Binary Output objects, due to common usage in the industry.

A BACnet automation station demo kit provided by SAUTER containing multiple Analog Input/Analog Output/Binary Input/Binary Output objects is used for the project. Figure 8 shows the basic configuration of the BACnet automation station which is taken into operation.

6. BACNET4J LIBRARY

6.1 ABOUT THE LIBRARY

BACnet4J stands for BACnet/IP for Java. It is an open source BACnet stack. As described by the project team of this library from the BACnet4J developer website, it is "a high performance implementation of the BACnet/IP protocol written in Java (minimum version 1.5) by Serotonin Software. It Supports all BACnet Services and full message segregation and can be used for field devices or for control platforms" [11].

6.2 FUNCTIONALITIES

The BACnet4J library enables the user to program an application communicating via the BACnet protocol using BACnet services. Most of the BACnet objects and properties (for example Analog Input, Analog Output, Binary Input, Binary Output and Device used in this project) can be

simply defined as Java objects. The BACnet services can also be created as a Java object, which represents all services and calls the methods from the library to execute the services.

With an additional programming on the OPC UA server side (in Java) the server is able to retrieve BACnet information from the BACnet devices using BACnet4J.

7. JAVA BASED OPC UA SERVER SDK

Java is the preferred language for the development of OPC UA server applications due to its platform independence. In addition, an open source library to communicate in BACnet (BACnet4J) already exists. Of course, other programming languages could also be used.

Prosys PMS Ltd. [12] is a contributor to the OPC Foundation UA Java stack and has developed an OPC UA Java SDK. This comes with both OPC UA server SDK and OPC UA client SDK. It is now available as a full release for purchase. Nevertheless, an evaluation edition is available. This project used the evaluation version of this OPC UA server SDK.

The Prosys OPC UA server SDK supports the creation of an OPC UA address space and information model in the server which is mainly required for the integration of the BACnet information (objects and properties) to the OPC UA nodes and variables.

8. BACNET-TO-OPC UA INTEGRATION

The first step in this integration process is to have all of the devices and systems in the same network domain, where all are accessible at least up to the network layer. For demonstration purposes, a local private network has been created and the devices are connected as illustrated in Figure 9.

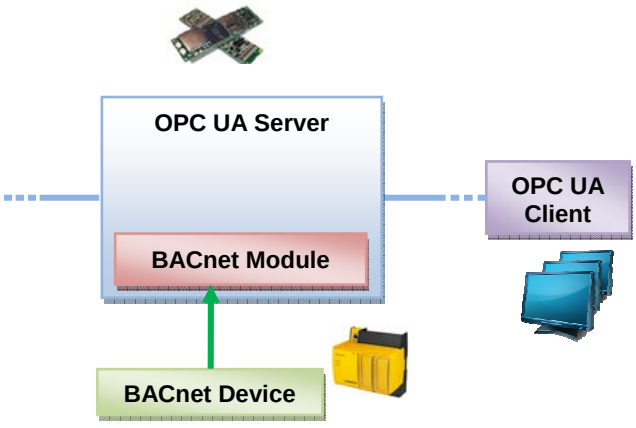


Fig. 9 – Connections of the devices

The BACnet Module inside the OPC UA server is the Java program that performs BACnet services (Remote Device Management and Object Access services) and stores the BACnet information in Java objects (BACnet4J functionality).

Based on the work of "*Interoperability at the Management Level of Building Automation Systems: A Case Study for BACnet and OPC UA*" [1] and "*Information Modeling in Heterogeneous Building Automation Systems*" [2], OPC UA node types in the list below must be prepared for the BACnet information:

- DataType
- ReferenceType
- ObjectType
- VariableType

In our project, we focused on the 'Present_Values' (cf. ch. 5.3) plus the concept of making the OPC UA information model for BACnet information as simple as possible. Consequently, the procedures in this project are the following: Starting with the 'DataType', OPC UA built-in data types can be used for BACnet 'Present_Values' directly without any issue. The data type definition of *Boolean* inherited from *BaseDataType* can be used for the BACnet 'Present_Values' of Binary Input and Binary Output and the data type definition of *Float* which is inherited from *BaseDataType»Number»Float* can be used for the BACnet 'Present_Values' of Analog Input and Analog Output. For all references of the nodes for BACnet information which will be stored in the *RefferenceType* folder of the OPC UA server, the reference type definition will be the OPC UA built-in *HasComponent* which is derived from *HierarchicalReferences »Aggregates»HasComponent.*

Some nodes of object type definition must be defined for BACnet objects as illustrated in Figure 10.

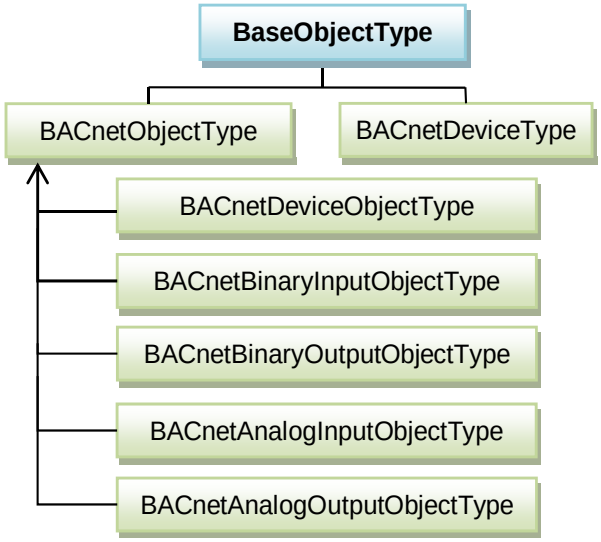


Fig. 10 – OPC UA Object Type Nodes for BACnet Objects

Every node of object type definition for BACnet objects (in green) inherits from the original OPC UA built-in '*BaseObjectType*' (in blue) like other OPC UA nodes of object type definition.

As well as ObjectType, some nodes of variable type definition must be defined for the BACnet properties. Figure 11 shows the node hierarchies for BACnet properties.

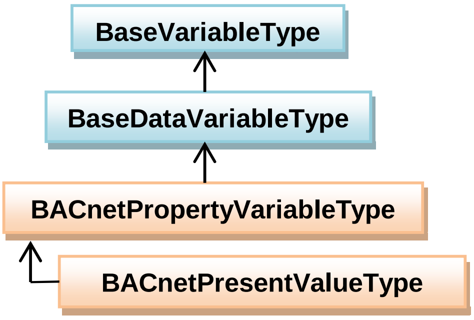


Fig. 11 – OPC UA Property Type Nodes for BACnet Properties

Every node of property type definition for BACnet properties (in orange) inherits from the original OPC UA built-in "*BaseDataVariableType*" and "*BaseVariableType*" (in blue). Other BACnet properties like "*Present_Value*" can be defined under the "*BACnetPropertyVariableType*", as well.

9. RESULTS

9.1 BACNET BROWSING RESULT

The results of browsing (discovery and reading properties) the BACnet demo device using the BACnet4J library, displayed in the console are shown in Figure 12.

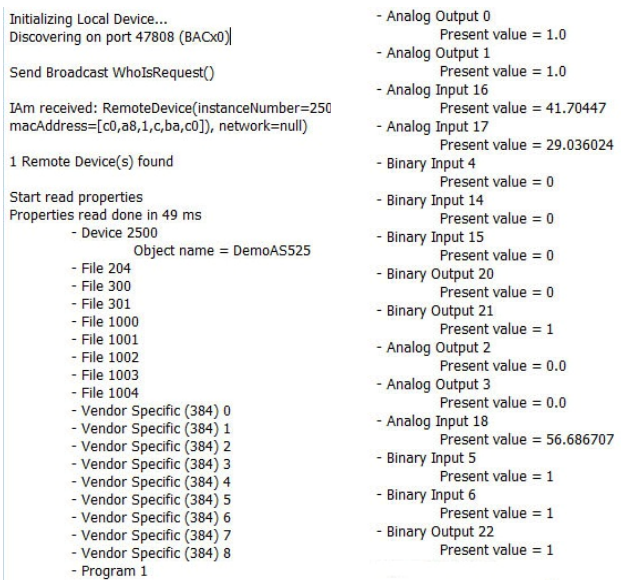


Fig. 12 – Results of Browsing BACnet Device

Figure 13 shows the BACnet packages monitored between the BACnet4J client and the BACnet automation station in wireshark.

9.2 BACNET IN OPC UA MODEL

With the result of browsing BACnet information in subsection 8.1 mapped to the OPC UA information using the procedure described in ch 7, the final results are stored in an additional folder 'BACnetObjects'. This result is displayed in a freeware OPC UA client 'UaExpert' from Unified Automation by browsing to the OPC UA server [13] as shown in Figure 14.

The OPC UA 'ObjectTypes' and 'VariableTypes' created for BACnet information, can also be displayed in the OPC UA client as displayed in Figure 15 and Figure 16.

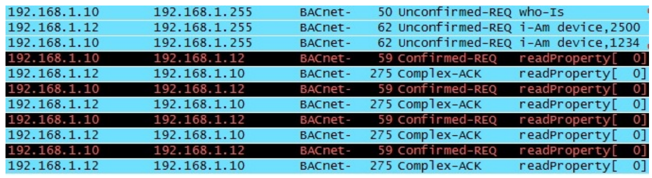


Fig. 13 – Packet Transfer Displayed by Wireshark

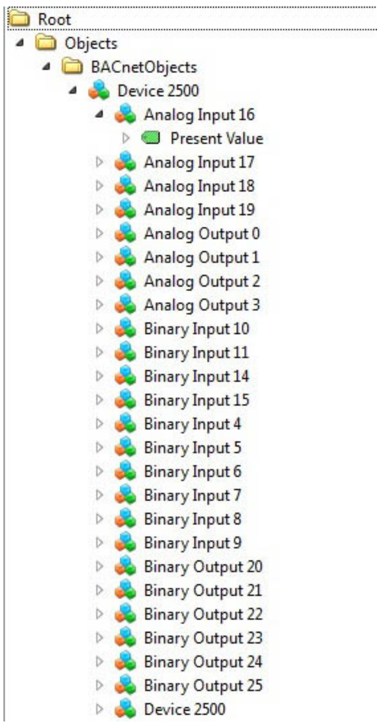


Fig. 14 – Result at OPC UA Client displayed by UaExpert

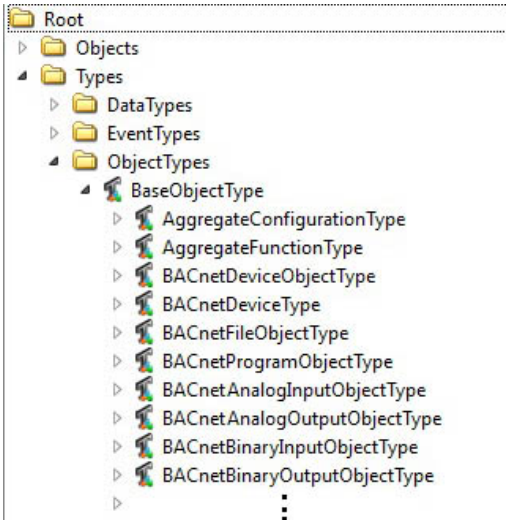


Fig. 15 – Result of OPC UA ObjectTypes for BACnet displayed by UaExpert

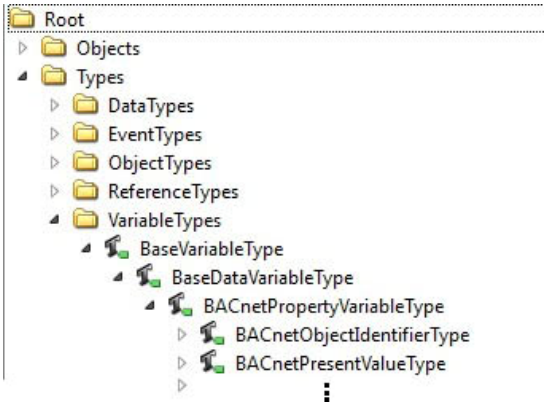


Fig. 15 – Result of OPC UA VariableTypes for BACnet displayed by UaExpert

10. CONCLUSION

Interoperability of the devices between different protocols allows significant benefits for the building automation industry. This paper shows an approach of how to integrate BACnet devices to OPC UA, which can be adapted to many use cases in the industry. Even tough, the result of this work might not cover all of the BACnet mandatory properties it shows that the integration can be done on embedded devices with restrictions in memory and processing power. Since most of the modern embedded devices are now much more vigorous than the old embedded systems. The integration task is done much faster, easier and without any issues.

Not only BACnet information can be integrated to OPC UA information model since OPC UA built-in can support more complex data models so it is also promising and feasible to integrate other devices from other protocols. Especially the ones that have a concept of object oriented data model.

11. REFERENCES

[1] A. Fernbach, W. Granzer, W. Kastner, *Interoperability at the Management Level of Building Automation Systems: A Case Study for BACnet and OPC UA*, IEEE ETFA, 2011.

[2] W. Granzer, W. Kastner, Information Modeling in Heterogeneous Building Automation Systems, *IEEE International Workshop*, 2012.

[3] OPC Unified Architecture Specification 1.01 Part1: Concepts and Overview, Part 3: Address Space Model and Part 5: Information Model, *OPC Foundation*, 2009.

[4] ASHRAE SSPC 135 BACnet Official Website. http://www.bacnet.org/.

[5] S.T. Bushby, BACnet: a standard communication infrastructure for intelligent buildings, in: *Automation in Construction* 6, Elsevier, pp. 529-540.

[6] *Information Processing Systems – Open Systems Interconnection – Basic Reference Model*. ISO 7498, 1984.

[7] B. Swan, The Language of BACnet, in: *Engineered Systems*, (13) 7 (1996), pp. 24-32.

[8] Gumstix Overo AIR COM specifications Sheet, Gumstix Offical Website. https://www.gumstix.com/store/product_info.php?products_id=226.

[9] Gumstix Tobi specifications Sheet, Gumstix Offical Website. https://www.gumstix.com/store/product_info.php?products_id=230.

[10] Sauter EY-modulo 5 Product Data Sheet, Sauter AG Website. http://www.sauter-controls.com/pdm/docs/en_ds_en690613.pdf.

[11] BACnet/IP for Java Developer Website, Serotonin, 2008; http://bacnet4j.sourceforge.net/

[12] Prosys OPC Official Website http://www.prosysopc.com/

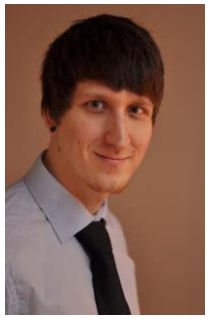
[13] Unified Automation Official Website http://www.unified-automation.com/

***Naksit Anantalapochai**, born in Brussels Belgium in 1988, graduated Bachelor of Computer Engineering from Faculty of Engineering, Chiang Mai University, Thailand in 2011. Since then, he is studying Master degree of Science "Communication and Media engineering" from University of Applied Science Offenburg, Germany. He has a high interest in modern innovations and technologies especially in the field of embedded systems, communication networks and computer science altogether.*

Prof. Dr.-Ing. Axel Sikora *holds a diploma of Electrical Engineering and a diploma of Business Administration, both from Aachen Technical University. He has done a Ph.D. in Electrical Engineering at Fraunhofer Institute of Microelectronics Circuits and Systems, Duisburg. After various positions in the telecommunications and semiconductor industry, he became a professor at the Baden-Wuerttemberg Cooperative State University Loerrach in 1999. In 2011, he joined Offenburg University of Applied Sciences, where he holds the professorship of Embedded Systems and Communication Electronics.*

His major interest is in the system development of efficient, energy- aware, autonomous, secure, and value-added algorithms and protocols for wired and wireless embedded communication. He is founder and head of Steinbeis Transfer Center Embedded Design and Networking (sizedn) since 2002.

Dr. Sikora is author, co-author, editor and co-editor of several textbooks and numerous papers, as well as conference committee member to various international conferences in the field of embedded design and wireless and wired networking.



David Eberlein *graduated as Bachelor of Engineering in Information Technology at the Cooperative State University Lörrach in 2009, winning the prize of the city Lörrach for the best bachelor thesis of the year. Since then, he has been working for Fr. Sauter AG as Software Engineer at the Technologies department.*

His main interests are web based graphical user interfaces, as well as mobile applications.

Dominique Stephan Kunz *studied electrical engineering (with a focus on control and automation engineering) at the University of Applied Sciences in Basel. Since 1999, he has been working for the company Fr. Sauter AG in Basel. He has over 10 years of experience in industrial product development and project management in the heating, ventilation and air conditioning market with respect to building automation.*

Currently, he heads the Technologies department, which deals with research projects, and he is responsible for cooperation with universities.