Semi-automated Engineering in Building Automation Systems and Management Integration

Andreas Fernbach and Wolfgang Kastner

Vienna University of Technology
Institute of Computer Aided Automation
Email: {afernbach, k}@auto.tuwien.ac.at

***Abstract*—Building planning as well as automation systems engineering involve numerous disciplines participating in a many-phase process. However, the information transitions between these different phases are mostly carried out manually which results in multiple sources of errors. Additionally, information which is already present is repeatedly manually re-entered during the whole workflow. This especially applies for building automation system engineering where on the other hand the majority of necessary information is already generated during the planning phase. To address this issue, this work proposes a workflow of semi-automatable engineering for building automation systems and also management gateway integration.**

I. Introduction

Today's lifecycle of a building normally starts with following consecutive phases involving a considerable number of different disciplines:

- strategic planning
- preliminary studies
- project planning
- construction bidding
- realisation
- operation, including maintenance and retrofitting

The information flow between these different phases is usually not carried out in a consistent way leading to information loss and misinterpretations finally resulting in severe problems and delays during the realisation phase. Initial requirements to the building physics and building services are fulfilled incompletely and thus construction and integrator companies are confronted with long lists of deficiencies.

In this context, one of the most affected trade is the integrators of technical building equipment. The information flow from building services planning to automation systems engineering involves extensive manual information transfer processes. Automation systems planning data needs to be interpreted by the engineers and technology specific engineering tool projects for device and system configuration need to be set up accordingly. This task involves repetitive and tedious steps which are error prone and time consuming. Additionally, many planning offices and integrator companies use their individual designation systems for automation systems and components which often provide only low expressiveness and therefore involve the danger of ambiguous systems description. This is again an error source during planning and engineering. It also exacerbates vendor independence and later third-party integration of example for data monitoring applications.

The goal of this work is to describe a process of continuous and machine readable information flow from building services planning to automation system engineering and also management gateway integration. Planning information shall be - as far as possible - entered once and reused during the whole workflow.

There have already been published concepts addressing this question. The most elaborated concepts in this field are [1] and [2]. They describe a multi-tiered method of automated design. Main attention is turned on field bus and automation systems but these approaches have a weak focus on the integration level. There exists also a commercial tool providing a data exchange interface between CAD programs and the KNX engineering tool ETS. This so-called "ETS APP PROJECTDATAEXCHANGE" from IT Gesellschaft für Informationstechnik mbH[1] also by its nature focuses on the field and automation level.

In academia, also the LINC middleware [3] plays a significant role. It addresses heterogeneous building automation systems integration and reconfigurable applications. Field device attributes are in this approach abstracted and encapsulated using a "bag" mechanism. Hereby, a technology-independent layer is generated where one can set up applications without knowledge about the underlying systems. Also the implementation of user interfaces is facilitated. A number of publications build upon the LINC middleware, like [4] (additionally utilising BASont [5], a building automation systems ontology) and [6]. A different middleware approach is described in [7], where a technology-neutral, OSGi-based intermediate layer is extended by application-specific DPWS, UPnP and HTTP service modules using a model driven engineering (MDE) process.

II. State-of-the-art technologies in building automation

KNX [8] is a widespread protocol in home and building automation systems. It finds a use especially in lighting applications but in other trades like HVAC as well. The protocol defined by KNX can be seen located at the field

[1] http://www.it-gmbh.de/

978-1-5090-1412-5/17/$31.00 ©2017 IEEE

level of the automation pyramid. It is characterised by narrow-bandwidth-communication which takes place on twisted pair line, RF, powerline and recently over IP-based networks. The interworking model of KNX relies on functional blocks (FBs) which are defined by the standard. They give an abstract description of the behaviour of the devices. Correct interfacing between FBs is enabled by the definition of distinct data point types (DPTs). By the logical connection of multiple devices being functionally compatible to each other (e.g. a dimmer and a light actuator) the desired distributed functionality of a KNX building automation system is achieved. This logical connection is established by a group addressing scheme. The configuration of devices necessary for that can be done by means of using the E-Mode which is an on-device configuration procedure via push buttons, or the S-Mode using the ETS as a central engineering and diagnostic tool.

OPC Unified Architecture [9] is an open standard published by the OPC Foundation [2] with the goal to establish interoperability between devices and networks of different technologies and vendors. It is the successor of the Classic OPC standard which is still widely used in industry. OPC UA encompasses significant improvement compared to the classical OPC, mainly with respect to data transport mechanisms based on Web Services following a client-server-architecture and comprehensive data modelling capabilities [10]. Within the so called OPC UA address space which is based on a system of nodes and references, information models can be instantiated. An information model can be seen as a description of how process data is represented in a distinct technology. In OPC UA such a model follows an object-oriented approach including well-known principles like type hierarchies and inheritance. An OPC UA server exposing an information model to an OPC UA client enables the latter to access process data of the underlying technology without the necessity of having any knowledge about its protocol specifics and the organisation of data. The goal of OPC UA is to define information models for the variety of different technologies deployed in a modern automation system. This way, a unified and consistent view to process data is provided to an OPC UA client, independent of the underlying technologies. Following this idea, OPC UA is usually applied at the management level of the automation pyramid where data are aggregated, archived and visualised and also where central access to setpoints takes place. Meanwhile, a number of information models for automation technologies have been included into OPC UA as so-called companion specifications, like OPC UA for Devices (OPC UA DI) [11] or OPC UA for IEC 61131-3 (PLCopen) [12] and BACnet [13].

Building information modelling (BIM) [14] is a concept of a common information base throughout the whole lifecycle of a building. One popular BIM format is [15]. It is designed as a unified data exchange format between CAD and design tools and tools for energy simulation. The schema defines a great variety of parameters concerning the building geometry, its architecture and physical properties. Attributes are hierarchically organised and provide attributes for sites, buildings, storeys, materials, rooms, surfaces and openings like doors and windows. Structural information is enriched with properties of building physics, e.g., albedo, glaze, emittance and U-value. An XSD schema formalises and explains all available parameters in gbXML and ensures compatibility between the different software tools.

IEC 81346 "establishes general principles for the structuring of systems including structuring of the information about systems. Based on these principles, rules and guidance are given for the formulation of unambiguous reference designations for objects in any system. The reference designation identifies objects for the purpose of creation and retrieval of information about an object, and where realized about its corresponding component."[16] Following these structures, ISO/TS 81346-3 [17] further defines rules how to form reference designations in order to uniquely identify an object within a plant or a building. This results in a systematic naming convention providing information about an embedded object's functionality, location and product related aspect. The benefits of such a reference designation system are the easy identification of systems and their parts and the use of a common language for aligning information about a whole plant containing of an enclosing building structure and the technical equipment. A reference designation describes one up to three of the previously called aspects or even additional, user defined ones. Syntactically, each one is signed by a prefix, "=" for the functional aspect, "-" for the product aspect and "+" for the local aspect. Standardised identification letters following the prefixes are used to classify the intended application of an object. The location can be signed by an individual systematic. Considering a motor (M01) driving a pump (G01) of a cooling system (E03) located in room number 4 of building A, the reference designation would look like the following: `-E03-G01-M01+A+4` where the hierarchical structure is represented in a top-down manner.

III. An OPC UA information model for KNX

In order to close the gap between KNX installations and the IT world by using Web services, a standardised interface has been defined. It allows communication between both systems by means of a gateway device implementing this interface. This section focuses on the OPC UA part of the so called and not-yet-released *KNX Web services specification* which has been developed by the TU Vienna Automation Systems Group in cooperation with the KNX Association[3]. However, [18] gives a summary of the specification.

The general approach of this KNX WS gateway concept is to expose the necessary pieces of information inherent in a KNX network to local or remote client devices using Web service technologies. This information includes

- building topology,
- device model (properties, meta information),

[2] www.opcfoundation.org

[3] https://www.knx.org

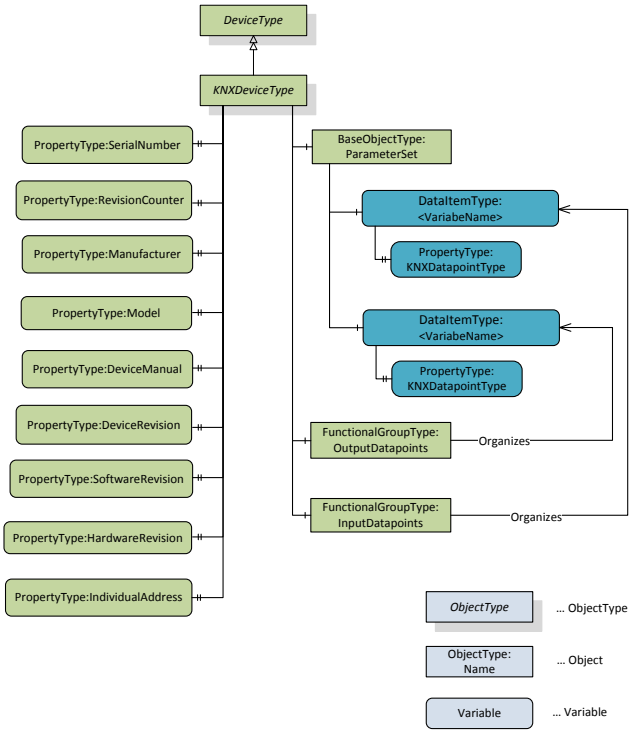


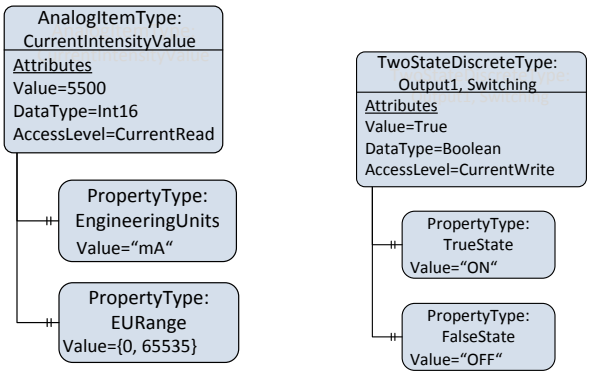
Fig. 1. OPC UA KNX device model

- live datapoints, i.e., the process image of the underlying KNX installation.

It is further enriched with semantics including a type system and physical units. It is intended to gain this information from the KNX engineering tool ETS in a mostly automated way. To this aim, ETS projects which in most cases exclusively contain the complete engineering information of a KNX installation are extracted. Exceptions thereof are KNX devices which rely on vendor-specific ETS plugins for configuration. The concepts of the KNX WS specification are very closely related to the way how information about a KNX installation is represented in the ETS.

The KNX WS specification defines an abstract tag vocabulary-based meta model with the ability to express the information mentioned above. It is independent of an actual technology implemented by an actual KNX WS gateway. In a second step, transformation rules from this meta model to the information models of the most common integration technologies OPC UA, OBIX and BACnet/WS models are defined. By this way, the KNX WS specification stays highly maintainable and extensible since changes only have to be applied to the tag vocabulary. These changes are automatically propagated to the derived Web service technologies by the defined transformation rules.

The OPC UA base information model provides a number of modelling elements (so called NodeClasses) acting as building blocks for higher-level or application-specific information models. It defines type definition elements like ObjectType, VariableType, DataType NodeClasses as well as the according



(a) AnalogItem (b) TwostateDiscreteItem

Fig. 2. Datapoint example

instance NodeClasses Object and Variable. Links between these Nodes are established by the ReferenceType Nodeclass. The usual modelling practice in OPC UA is to design type definitions of repetitive patterns like devices and datapoints in a first step which are then instantiated in the address space of an OPC UA server (gateway). Complex type definitions are very common like for example an ObjectType referencing a number of Variables (datapoints). For the OPC UA incarnation of the KNX WS metamodel, the topological aspect of the KNX information model is reflected by an arbitrary deep hierarchy of the built-in ObjectType `FolderType`. By means of such a structure of folders it is possible to reflect a building layout consisting floors, rooms and stairways. For the device and datapoint aspects of a KNX installation it is drawn on already published high-level OPC UA information models. The already mentioned OPC UA for Devices (DI) specification provides a very detailed concept for modelling field devices and is therefore well-suited for the KNX device model. Figure 1 shows the abstract type definition of a `KNXDeviceType` which is a subtype of the abstract `DeviceType` defined in the DI specification. It therefore inherits its structure from its supertype which is extended by KNX-specific properties. The properties on the left branch of the `KNXDeviceType` definition provide static information about the device like a serial number, hardware or software revision or manufacturer information. The `IndividualAddress` Property is an extension to the `DeviceType` and reflects the physical, i.e., individual address of a KNX device. The `ParameterSet` Object on the right side exposes the actual datapoints of the device. Their number and types naturally vary for each concrete KNX device for which this abstract definition provides the necessary freedom. A `KNXDatapointType` property informs about the underlying KNX datapoint type (DPT). The `Output-Datapoints` and `InputDatapoints` Objects assign an I/O direction to the datapoints.

In order to model the semantic of KNX datapoints defined in their according type definition (KNX DPT) the OPC UA Data Access model [19] is used. It defines a number of gen-

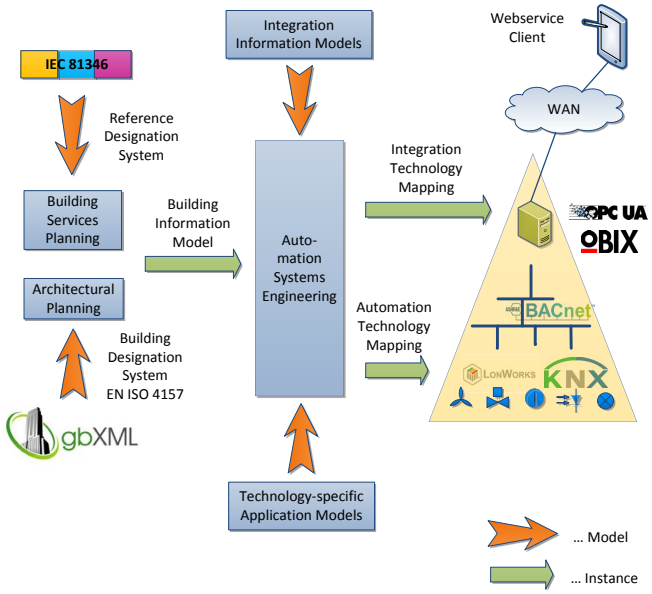


Fig. 3. Engineering workflow

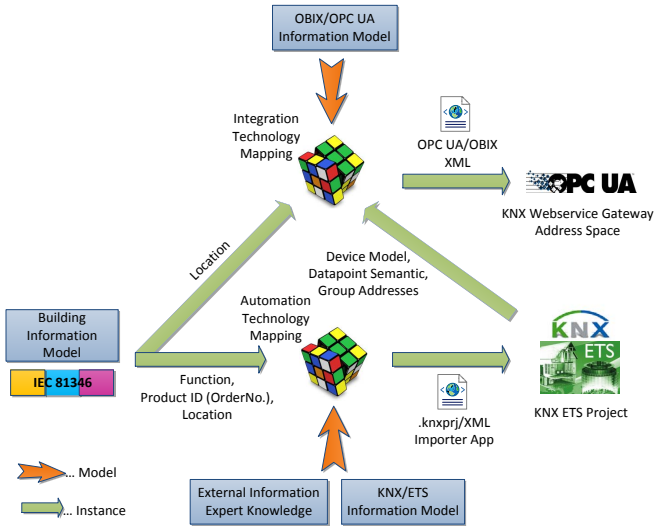


Fig. 4. Automated engineering

eral purpose VariableTypes for modelling analog and discrete datapoints. Besides the actual value, additional semantics like datatype, access level, unit and value range are provided. Figure 2 shows two instance examples of a `CurrentIntensityValue` and a switching value `Output1, Switching` datapoint. When instantiating a concrete KNX device, DataItems like shown in these examples are inserted beyond the `ParameterSet` Object of the `KNXDeviceType`.

IV. Semi-automated engineering

The general information flow of planning a building and its automation system components to the engineering at automation level and integration level is shown in Figure 3. The orange-colored arrows indicate the application of models, i.e., the patterns behind the instances generated during this workflow. Models in this application field can be BIMs, standards defining designation systems for buildings (like EN ISO 4157 [20]), technical equipment (IEC81346) or technology-specific application model like for KNX or OPC UA. During the planning and engineering process of a building, the actual problem is formulated and instantiated applying these models at multiple states of the overall progress.

In the beginning, when the building owner, architects, civil engineers and building automation engineers carry out the planning phase the use of tools supporting BIM standards like gbXML and designation systems like EN ISO 4157 and IEC81346 reduces error sources and improves the maintainability and reliability of produced planning data. Examples for such tools are the widespread products *AutoCAD Architecture* and *Eplan Electric*. This resulting planning data, which can be seen as a *holistic building information model*, includes structural and geometrical information of the building as well as structured information about the technical equipment. The goal is to include as much as possible of the information needed in following engineering phases already at this point. This is the prerequisite for implementing (semi-)automated technology mapping processes when it comes to automation systems engineering.

The pyramid-shaped object on the right side of Figure 3 shows the typical structure of building automation systems. At the bottom field devices interacting with the physical processes in the building. These devices usually communicate via field bus protocols like KNX or LonWorks [21]. BACnet [22] devices are usually used at automation and management level. Multi-protocol integration and gateway interaction is also located at the management level in top of this pyramid model of automation systems.

The planning information previously generated (i.e., the holistic information model) constitutes the information source for the next step, namely for mapping abstract building services to specific application models of building automation technologies. This process is normally understood as automation systems engineering. Planning data is manually processed and put into technology-specific and in most cases (except KNX which uses the ETS) vendor-specific engineering tools. This process is carried out on multiple layers of automation systems. Initially for field and automation technologies but later or even in parallel when it comes to SCADA integration and gateway engineering on top level of automation systems. However, parts of this process can be automated utilising information from properly structured planning data.

Figure 4 shows this technology mapping process in detail. It takes planning information consisting of structural building information and automation systems description using the IEC 81346 reference designation system. Hereby, following aspects are included regarding automation equipment:

- Location

- Function
- Product ID (order number)

The location aspect shares a common designation system with the building information model and therefore, a unique assignment of automation equipment to a topological building element is assured. The KNX ETS technology mapping (engineering) relies on the product aspect and the location aspect, whereas the functional aspect is optional. In an ETS project, KNX devices are uniquely identified by their order number. This allows adding KNX devices according to the list of IEC81346 reference designations to an ETS project in a straight forward manner. In a next step, the building view in the ETS is instantiated with building topology defined in the BIM. Following the shared building designation system, KNX devices are assigned to their intended building topology elements. This results in an ETS project skeleton which can be automatically generated. The ETS App API allows interacting with the ETS via third-party applications providing this information to the ETS. Still, there are parts of the engineering process that need to be carried out manually. On the one hand proper configuration of devices by setting their parameter requires information which is not available by the proposed planning data. Moreover, the definition of functional relationships on the level of datapoints which is achieved by the definition of communication groups is not automatable using this concept since the designation systems in use do not support behavioural modelling. AutomationML[4] might be considered to fill this gap. This group communication engineering is therefore a remaining manual step which is in practical ETS project engineering carried out by assigning shared group addresses to the intended communication partners.

In order to instantiate a KNX WS gateway according to the planning information, additional information is needed from the KNX ETS project. The planning information does not include a device model giving a description of the set of datapoints a device exposes. Also the datapoints' semantics is missing, i.e. the datatype, engineering unit and value range. However, this information is contained in the device model of the ETS. The datapoint types where the ETS device model refers to are exactly defined in the KNX System Specification. They include the desired meta information like datatype, encoding, unit and range. In Figure 4, this information flow is illustrated by the upwards-heading arrow from the ETS symbol to the integration technology mapping symbol. Additionally to that, also the group address scheme assigned to a KNX installation needs to transferred to such a gateway. Like a native KNX device, it needs the address information to be able to properly access KNX datapoints in the network.

Hereby, the necessary information to instantiate a KNX WS gateway using OPC UA is available. The topological structure can be directly gained from the BIM which is in a further step linked to the device-related information. The result can be formalized following the standardised OPC UA XML schema definition (XSD) and loaded as an XML configuration file into the gateway implementation.

This engineering process concept is illustrated in the following by an example. A model of a building is assumed to have four floors whereas the fourth floor contains two rooms, an office room and a meeting room. This topological elements are labeled using a designation system like the following: Building `DE`, fourth floor `04`, office `23` and meeting room `22`. This results in the unique designations `+DE+04+23` for the office and `+DE+04+22` for the meeting room.

The gbXML code snippet for the Office room is:

```
<Space id="+DE+04+23" conditionType="
    HeatedAndCooled">
        <Name>Office</Name>
                ...
```

and for the Meeting room:

```
<Space id="+DE+04+22" conditionType="
    HeatedAndCooled">
        <Name>Meeting Room</Name>
         ...
```

The office is equipped with a switching actuator assigned with the IEC81346 reference designation

=EA=KF03−5WG1562−2AB31+DE+04+23

The functional aspect of this device is declared with the hierarchical descriptor `=EA=KF03` standing for a controller device being part of a lighting system. This is followed by the product aspect `-5WG1562-2AB31` representing the ETS order number of the device. `+DE+04+23` denotes the local aspect which conforms to the label of the room where the switching actuator is located. Analogously to that, a temperature controller is denoted with

=EP=KF01−5WG1253−2AB_3+DE+04+22

where `=EP=KF01` indicates the functionality of a controller device as part of a heating system. `-5WG1253-2AB_3` stands for the order number of the device and `+DE+04+22` for the local aspect.

A. KNX ETS engineering

Following the workflow concept described in the previous section, the information from the exemplary building model and from the device designations is taken - with respect to the possible degree - to generate an ETS project skeleton. The screenshot in Figure 5 shows a snippet of the resulting ETS project configuration. The ETS *Buildings* view exposes the hierarchical building topology of the building `+DE` with the fourth floor `+DE+04` and the two rooms `+DE+04+22` and `+DE+04+23`. The strings representing the human readable names of the topological objects (`4th Floor`, `Meeting Room`, `Office`) are taken from the gbXML building model. Beyond these two room elements, the respective KNX devices are instantiated. This assignment is achieved by the order number given by the device designations.

[4]AutomationML association, https://www.automationml.org

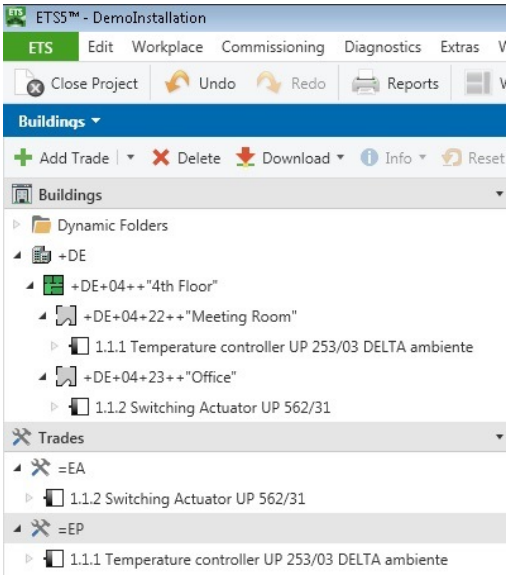


Fig. 5. ETS project

Additionally to the Buildings view, the ETS provides a perspective called *Trades* where devices can be arranged according to their functionality. This view is used to reflect the functional aspect denoted by the devices' reference designation system. Hence, two trades are instantiated, `=EA` for lighting `=EP` for heating. Then, the exemplary KNX devices are assigned to these items.

B. KNX WS/OPC UA gateway engineering

The following phase of automation systems engineering is about to instantiate a management gateway providing Web service access to the underlying installation. In this work, the focus lies on integration by means of OPC UA using the new KNX WS specification. Like for the ETS engineering process, already existing information is processed and used to create OPC UA instances. This also includes information derived from the previously generated ETS project. The resulting OPC UA instance which is compliant to the KNX WS specification is shown in Figure 6. The `Root` object is the entry point for an OPC UA client exploring the address space of a server. A real OPC UA server exposes a number of folders (containing e.g., ObjectTypes, VariableTypes, ReferenceTypes) at this point which are not shown in this figure except for the `Objects` folder. The set of instance Nodes of a server are grouped beyond this item. The orange-colored hierarchy of `ViewType` objects reflects the building topology. It terminates with the two example rooms `DE+04+23++Office` and `DE+04+22++Meeting Room`. Via `Organizes` ReferenceTypes, two Objects representing KNX devices `KNX-TemperatureController` `KNXSwitchingActuator` are linked. These two device objects can be additionally browsed via the `DeviceSet` object which is required by the OPC UA DI companion standard. The KNX WS specification includes this device model as already mentioned. For simplicity, only the switching actuator device shows part of

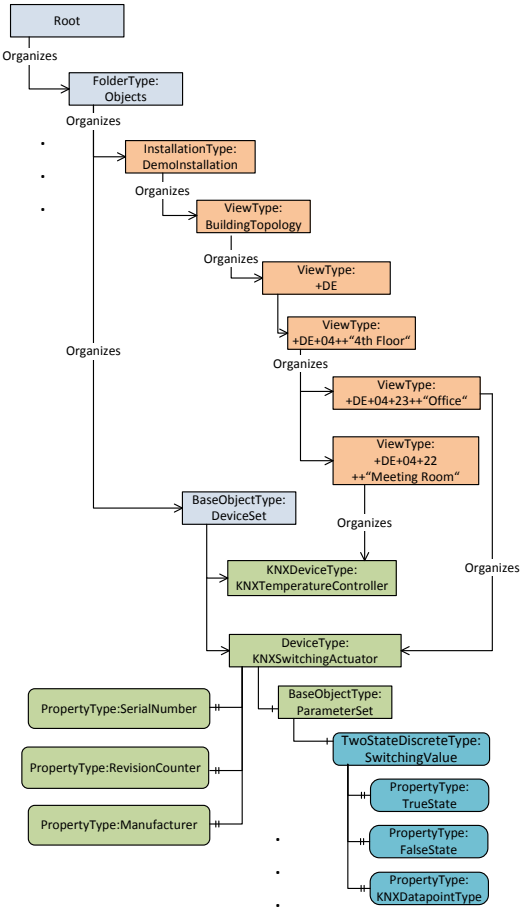


Fig. 6. OPC UA Instance

its address space. The `ParameterSet` object exposes a binary datapoint variable `SwitchingValue` which is enriched with datapoint semantics about the encoding (`TrueState` `FalseState` properties) and the originating KNX datapoint type (`KNXDatapointType`). As examples for static device information, three properties informing about the serial number, revision and manufacturer are displayed. The information necessary for instantiating the devices, their static properties and the datapoints originates from the ETS project. One aspect which is not visible in the OPC UA representation is the group addresses which are required for runtime interaction between the KNX Ws Gateway and the KNX network. The are also taken from the ETS project and saved internally in the gateway.

Figure 7 shows the runtime interaction between the gateway and an OPC UA client. When a client wants to read the current value of a Node attribute (e.g., the current value of a datapoint), it issues a *Read* service call to the KNX OPC UA Gateway. The gateway internally determines if the attribute to read is a KNX datapoint or statically defined value in the OPC UA servers address space (e.g., a DisplayName or a Description). If the read access regards a KNX datapoint, the gateway resolves the request to an internally stored KNX group address and puts a *Group Value Read* request to the

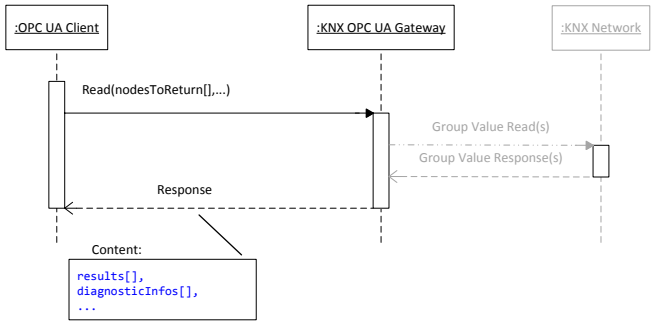


Fig. 7. Read transaction

KNX network. After the KNX network has responded, the gateway can return the OPC UA Read response containing the DataValue and additional diagnostic information to the client. This diagnostic information contains a time stamp, status information about the gateway or error codes in case the request did not success.

V. Conclusion and Outlook

In this work, a concept for use in semi-automated information processing in building automation engineering has been proposed. It defines a first approach of a consistent workflow from building and automation system planning information to the engineering tool of the popular building automation standard KNX. Furthermore, a way to automatically instantiate a KNX WS gateway based on this planning data and additional device and datapoint semantic from the KNX application model. The newly designed KNX OPC UA information model this gateway follows, also including runtime interaction has been briefly described.

Still, one drawback of the proposed approach is the remaining manual interaction part during ETS project engineering. This issue might be overcome by integrating the ideas from [1] at this point of the workflow. Therefore, additional research in this area is required. Also the implementation of prototypes performing the proposed automated engineering processes is ongoing work.

Another future capability is the automated generation of visualisations at SCADA level. To this aim, the current workflow needs to be extended towards user interface markup languages like defined by the W3C or available for Java applications.

Acknowledgement

This work was funded by FFG (Austrian Research Promotion Agency) under the project “Kognitive Regelstrategieoptimierung zur Energieeffizienzsteigerung in Gebäuden KORE”, FFG 848805.

References

[1] H. Dibowski, J. Ploennigs, and K. Kabitzsch, “Automated design of building automation systems,” *IEEE Transactions on Industrial Electronics*, vol. 57, no. 11, pp. 3606–3613, 2010.

[2] H. Dibowski, U. Ryssel, and K. Kabitzsch, “Ganzheitlicher, automatischer Entwurf drahtloser Gebäudeautomationssysteme,” *at Automatisierungstechnik*, vol. 61, 2013.

[3] L.-F. Ducreux, C. Guyon-Gardeux, S. Lesecq, F. Pacull, and S. R. Thior, “Resource-based middleware in the context of heterogeneous building automation systems,” in *IECon 2012-38th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2012, pp. 4847–4852.

[4] F. Pacull, L.-F. Ducreux, S. Thior, H. Moner, D. Pusceddu, O. Yaakoubi, C. Guyon-Gardeux, S. Fedor, S. Lesecq, M. Boubekeur *et al.*, “Self-organisation for building automation systems: Middleware linc as an integration tool,” in *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*. IEEE, 2013, pp. 7726–7732.

[5] J. Ploennigs, B. Hensel, H. Dibowski, and K. Kabitzsch, “Basont-a modular, adaptive building automation system ontology,” in *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2012, pp. 4827–4833.

[6] L.-F. Ducreux, C. Guyon-Gardeux, M. Louvel, F. Pacull, S. R. Thior, and M. I. Vergara-Gallego, “Rapid prototyping of complete systems, the case study of a smart parking,” in *Rapid System Prototyping (RSP), 2015 International Symposium on*. IEEE, 2015, pp. 133–139.

[7] G. Nain, E. Daubert, O. Barais, and J.-M. Jézéquel, “Using mde to build a schizophrenic middleware for home/building automation,” in *European Conference on a Service-Based Internet*. Springer, 2008, pp. 49–61.

[8] “KNX Specification,” ISO/IEC 14543-3, 2014.

[9] “OPC UA Specification, Release 1.03,” OPC Foundation, 2015.

[10] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC Unified Architecture*. Springer, 2009.

[11] “OPC Unified Architecture for Devices, Release 1.01,” OPC Foundation, 2013.

[12] “OPC UA Information Model for IEC 61131-3, Release 1.00,” PLCopen and OPC Foundation, 2010.

[13] BIG-EU and OPC Foundation, “OPC UA Information Model for BACnet,” 2016, public Review 0.17PR.

[14] C. Eastman, *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors*. Wiley, 2008.

[15] “Open Green Building XML Schema: a Building Information Modelling Soltution for Our Green World,” 2016. [Online]. Available: www.gbXML.org

[16] “IEC 81346-1:2009 - Industrial systems, installations and equipment and industrial products Structuring principles and reference designations Part 1: Basic rules,” 2009.

[17] “ISO/TS 81346-3:2012 - Industrial systems, installations and equipment and industrial products Structuring principles and reference designations Part 3: Application rules for a reference designation system,” 2012.

[18] D. Schachinger and W. Kastner, “Integration von KNX Netzwerken in das Internet der Dinge: Die KNX Web Services Spezifikation,” in *Tagungsband des 8. Jahreskolloquium Kommunikation in der Automation - KommA*, Dec 2016.

[19] “OPC Unified Architecture Specification Part 8: Data Access, Release 1.03 ,” OPC Foundation, 2015.

[20] “Construction drawings - Designation systems - Part 1 : Buildings and parts of buildings (ISO 4157-1 : 1998),” EN ISO 4157-1, 1998.

[21] “Open Data Communication in Building Automation, Controls and Building Management Control Network Protocol – Part 1-5,” ISO 14908-1 - ISO 14908-4, 2008.

[22] “Building Automation and Control Systems (BACS) – Part 5: Data Communication Protocol,” ISO 16484-5, 2016.