Building Management System and IoT technology: data analysis and standard communication protocols for Building Information Modeling

Beatrice Maria Toldo, beatricemaria.toldo@unipd.it
Center for Energy Economics and Technology Studies Giorgio Levi Cases, University of Padova, Italy
Carlo Zanchetta, carlo.zanchetta@unipd.it
Department of Civil, Environmental and Architectural Engineering, University of Padova, Italy

Abstract
The study focuses on the analysis of building management system (BMS), also known as building automation control system (BAS - BACS), which refers to a computer-based control system installed inside buildings to monitor and operate electrical and mechanical equipment. The development of new technologies has led to the emergence of smart buildings, that can collect data from devices and sensors using IoT (The Internet of Things) solutions. Thanks to Building Information Modeling (BIM), the successful adoption of these new digital technologies has been used increasingly in construction and facilities management. Despite the implicit integration of different systems, severe interoperability issues affect the integration of BMS operating on different fields. One often has to deal with various communication protocols and proprietary BMS platforms that use different communication modes. This document aims to analyze the data exchange between BMS and BIM through the study of file formats exported by the platforms to create an integration of IFC (Industry Foundation Classes) schema in BMS platforms, specifically using the BACnet protocol. In this way, it is proposed to use the existing standard (IFC) as a foundation for integrating communication between multiple platforms. This approach aims to streamline modes of communication and facilitate the creation of a digital twin.

Keywords: BMS, IoT, IFC, BACnet, Python

1 Introduction

The recent development of technology in AEC industry has enabled data acquisition from BMS devices through smart sensors. BMS systems, also known as BACS, are solutions installed inside buildings that control and monitor services responsible for heating, cooling, ventilation, air conditioning, lighting, etc. There are Building Automation Protocols, which are rules and standards that make it possible to communicate between different devices in building automation. BMS platforms are quite closed systems that use proprietary protocols, customized for clients where the vendor and system integrator are required for extensions and upgrades (McGibney et al. 2016). Control devices of different manufactures need gateways to be integrated, which can cause problems with different proprietary protocols. There is a need to use the same protocol to ensure interoperability and integration. This control network based on open protocols is called “open networks”. The control network uses local connection protocols: TCP/IP, HTTP. There is a need for an interface between these two protocols to integrate a local control network and the Internet. Through these technologies it is possible to communicate data in real time between BMS, local networks and the Internet (Wang and Xie 2002).

To date, open protocols have been developed, that enable interoperability, including: KNX, BACnet, DALI, DMX, Modbus, LonWorks.

The integration of BIM with BMS software makes possible the creation of digital twin, a virtual representation of a building designed to reflect a physical object accurately, that is updated with

real-time data and used for simulations. Despite this, there is still a gap in data integration between BMS and IFC (Industry Foundation Classes) standard.

The paper is structured as follows. Section 2 provides background information related to IFC, Communication standards and BACnet. Section 3 gives the methodology and details regarding the creation of a tool for IFC to serve as a common information database for BMSs. Section 4 discusses the results and limitations of the project. Section 5 provides conclusions and possible future developments.

2 Background

2.1 IFC, IDM, MVD

BIM model can be used to contain all device-specific information, including links to manuals, access credentials for reprogramming, and device interconnection information. Once the building is built, the BIM model is the first tool to help facility managers, as it contains all the information about the construction and all the knowledge about the operation and maintenance of the facilities, including the automation systems. BAS information does not require the entire IFC schema for data representation; only an IFC sub-scheme that corresponds to the BAS communication protocol is needed for the exchange of information between software and project phases. IFC is the most used data exchange format for open BIM and it has been accepted as ISO 16739 standard. IFC enables data exchange between different software applications across the entire building lifecycle.

To overcome the problem of exporting only the necessary data, buildingSMART International created the IDM (Information Delivery Manual) (buildingSMART International 2024a) and MVD (Model View Definitions) (buildingSMART International 2024b) approaches for the purpose of defining subsets of the IFC schema for certain ERs (Exchange Requirements or information exchange requirements). IDMs are standardized by ISO 29481-1 and ISO 29481-2. In 2012, buildingSMART published an integrated IDM/MVD approach called "An Integrated Process for Delivering IFC Based Data Exchange" which incorporates the IDM and MVD approaches (Davis et al. 2012). An IDM aims to specify the processes that exist during the life cycle of a certain processing and the sequence of information needed to develop them. Thus, it deals with standardizing BIM exchange processes and improving the quality of communication between the different actors involved in the project. BuildingSMART has published official MVDs, including COBie (Construction Operation Building Information Exchange), which is used to exchange specifications for the acquisition and delivery of information in facilities management. Although COBie acquires information related to the life cycle, the data entities in the COBie MVD documentation have focused specifically on the construction management domain.

A recent study (Tang et al. 2020) demonstrates the possibility of exchanging BAS information compliant with the BACnet protocol with the IFC data model for BAS design and operation between BIM tools and Facility Management tools in various project phases. It successfully leveraged IDM and MVD methodologies to define a subset of the IFC schema (a BACnet MVD) that represents BACnet object types and their property identifiers to assist BAS design systems using BIM authoring tools.

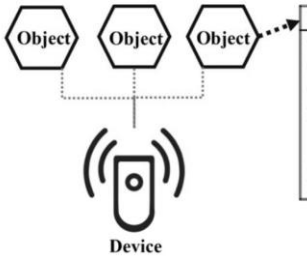
2.2 Communication standard

Different information protocols represent, distinguish and relate concepts of a specific domain by leaning on a diffuse domain model in which different applications are able to share information. Today, modeling data structures as objects, including control domain structures, is used. The open automation models used in building automation applications are BACnet, LonWorks, ZigBee and EIB/KNX.

2.2.1 BACnet

BACnet structure is distinguished from other communication technologies by a type of object-oriented organization of data, which have certain properties and guarantee the protocol to develop services. In BACnet, a device object is characterized by properties such as the model's

name, vendor, state, and list of other BACnet Objects associated with the device. Currently, there are approximately 55 Object Types (BACnet Object Types) defined in the American ANSI/ASHRAE 135:2016 standard (American Society of Heating Refrigerating and Air-Conditioning Engineers 2020). The instances of Object Type are Objects. A device can be represented as an aggregation of BACnet objects, each containing information about a component of a BACS system, such as a sensor or actuator. This information is grouped within property sets, which allow objects to be different and can be datapoints, i.e., a logical representation of data process. Each BACnet Object contains at least the following 3 properties: Object Identifier, Object Name, Object Type (Figure 1).



Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	R
…	…	…
Description	CharacterString	O
Device_Type	CharacterString	O

Figure 1. BACnet device, BACnet Objects and properties (Tang et al. 2020)

One of the standard BACnet objects is the analog input object (Figure 2), which represents an analog sensor input such as a thermistor. The BACnet standard identifies 123 different object properties, and for each type of object a different subset is specified. The purpose of a property is to allow other BACnet devices to read information about the object and potentially set a different value in the property. The specification requires that some properties must be present for each object, while others, specified by the manufacturer, are optional. The BACnet specification requires that some of the standard properties be mandatory set while others can be optionally set (Figure 3). In general, everything can be read by everyone.

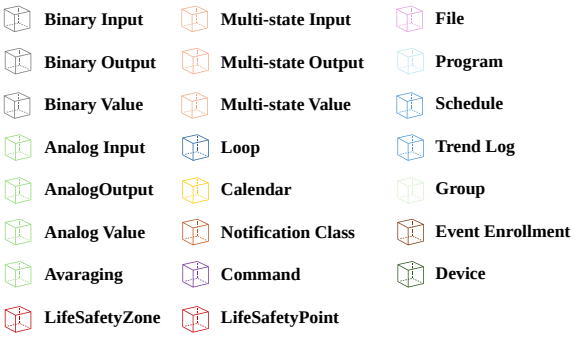


Figure 2. BACnet Object type

Required	Object_Name	SPACE TEMP
	Object_Type	ANALOG INPUT
	Present_Value	72,3
	Status_Flag	Normal, InService
Optional	Hight_Limit	78,0
	Low_Limit	68,0

Figure 3. Example of required property and optional property

2.3 Data model and standards

What stands out is that on one hand IFC can be considered both as a data model and an information source; on the other hand, there are many standards (BACnet, KNX, etc.). However, it is important to emphasize that the naming used to identify the entities in BMS is not crucial, because the item will remain the same, regardless of the used terminology. In addition to this, there are also BMS platforms that act as aggregators, which collect and transmit information, however, always in a proprietary format, using proprietary terminologies. In this scenario IFC

can act as a common basis for mapping elements, establishing an open-source database that can solve the problem at its origin (Figure 4).

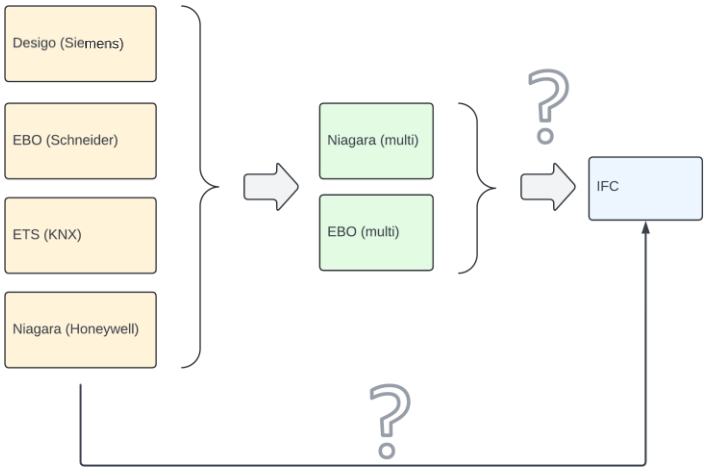


Figure 4. IFC as a common base for mapping BMS elements

3 Methodology

The literature review has led the experimentation in focusing on IFC mapping for BMS using a single file format. In the case study, experimentation is tested using an XML file, but the study cannot ignore to consider that different BMSs can export different file formats such as SQL. It is important to emphasize the dual approach of the methodology. On the one hand, the interoperability between open communication standards for both BIM (IFC) and BMS (BACnet, KNX, etc.); on the other hand, the plurality of file formats that can be exported. Although the most common is XML, this is not necessarily common for all BMS software. Exporting an SQL requires the creation of a different communication driver than an XML, even though the communication standard is, for example, BACnet in both cases. Thus, it is understood that the contrast between multiple file formats and a few data models brings out the belief that the data model (IFC) forms the basis of all logic for creating the information exchange.

One topic is to test what tools and solutions exist to ensure information exchange between IFC and BMS, which can be done in Visual Studio Code with Python, but also with any other coding language.

The study focuses on the importance of working on the data model and not on the file format. The latter is not of interest as it relates specifically to BMS platforms. Instead, it is very important to understand the great versatility of the IFC data model and its ability to operate as a common information exchange solution.

The study is in a prototype phase of analyzing and defining the technological aspects of information exchange. It aims to define the data model with the purpose, in the future, of compiling Exchange Requirements (ER) and Process Map (PM) that are the foundation of IDM. The analysis of ERs at this point is of little interest, as the work focuses mainly on the information and process technology aspect, not on the total mapping of the data model.

The application case study presented below is created with the aim of testing the systematic integration of an IFC file in the BACS environment using the BACnet protocol, reducing the burden, simplifying procedures, and removing the ambiguities arising from processing multiple models in different disciplines (Figure 5).

In the first part, there is an introduction to the BIM modeling of a prototype room containing the minimum devices needed to implement a small BMS. Special attention is paid to correctly exporting the various elements that conform to the IFC standard.

The second part deals with the analysis of a BMS platform export file to understand how the IFC hierarchy can be replicated within the BMS.

The last part focuses on using the IFC data structure to fill in information within a file to be imported into the BMS by creating Python code within Visual Studio Code.

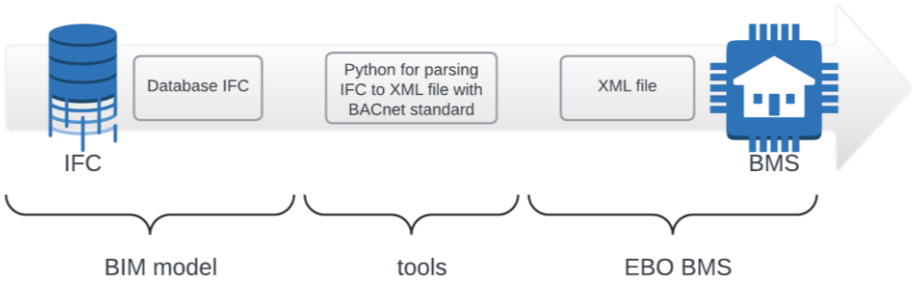


Figure 5. Methodology

3.1 BIM model

The BIM model consists of a room populated with all those objects aimed at the automated switching on of a light bulb: a light control, an occupancy sensor, an actuator, and the bulb itself, all connected to an electrical panel (Figure 6). The devices are all manufactured by Schneider Electric.

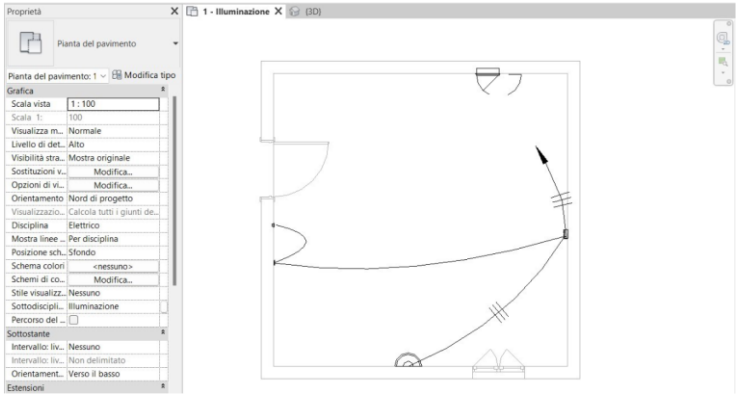


Figure 6. BIM model floor plan in Revit

3.1.1 IFC mapping

For each device was studied the IFC classification, populating entities with the appropriate properties (Table 1). The IFC4 Add2 standard was implemented (buildingSMART International 2020a).

In mapping the devices, their IFC class, TypeEnum, and IfcGuid are emphasized. The latter is a unique software-generated code assigned to each object, independent of its nature and not manually attributable by the user. It makes coordination of the subsequent specialized model easier since the components are distinctly recognizable.

Table 1. Model devices with associated IFC classification

Devices	Name	IFC Class	TypeEnum	IfcGuid
Light Control	KNX - System M - Push-button module - 1-gang	7.4.3.4.7 IfcSwitchingDevice	7.4.2.24 TOGGLESWITCH	1kaynQ0sLEOQzHVLIdScgm
Sensor	SE - SmartX - Optimum Housing - Touch Screen with Off-screen Control	7.2.3.9 IfcSensor	7.2.2.5 MOVEMENTSENSOR	2I4cCfk$10fPdL3W3bI4Oe
Actuator	KNX - Actuator – Switch – 10A - Manual mode	7.2.3.1 IfcActuator	7.2.2.1 ELECTRICACTUATOR	2I4cCfk$10fPdL3W3bI3hu
Bulb	M_Applique – Direct Lighting 120V	7.4.3.33 IfcLightFixture	7.4.2.17 POINTSURCE	2I4cCfk$10fPdL3W3bI3R_
Switchboard	Schenider-Electric Pragma flush mounted LV distribution board	7.4.3.13 IfcElectricalDistributionBoard	7.4.2.7 WITCHBOARD	2I4cCfk$10fPdL3W3bI4Bo

The domains to which the devices belong are specified:

- 7.2 IfcBuildingControlsDomain, to which sensor and actuator belong;
- 7.4 IfcElectricalDomain, to which the light control, bulb and switchboard belong.

On the official IFC schema page (4_ADD2 version), the IfcBuildingControlsDomain schema is part of the Domain Level of the IFC model (Figure 7). It defines the concepts of building automation, control, instrumentation, and alarm. The IfcBuildingControlsDomain schema (buildingSMART International 2020b) supports types and occurrences of:

- Actuator
- Alarm
- Controller
- Sensor
- Flow instrument
- Unitary control element

The IfcBuildingControlsDomain schema does not specify building automation protocols but can be mapped to standard protocols or vendor implementations for commissioning and interoperability of operations.

The IfcElectricalDomain (buildingSMART International, 2020c), also part of the Domain Layer, defines the concepts of wired systems in which wiring carries power supply, data, telephone signals or other forms of cable transmission. It also defines various devices that are connected by wiring, protection of electrical devices, supply and concepts of lighting fixtures within buildings, the wiring itself, and methods for supporting and carrying cables.

The IfcElectricalDomain scheme supports ideas, including types of:

- Audiovisual equipment,
- Cable chain fittings (for raceway, cable tray, and stairs)
- Household appliance
- Electric motor
- Distribution boards
- Generator
- Junction box
- Lamp
- Shop
- Protective device
- Protective device release device
- Switching device
- Transformer

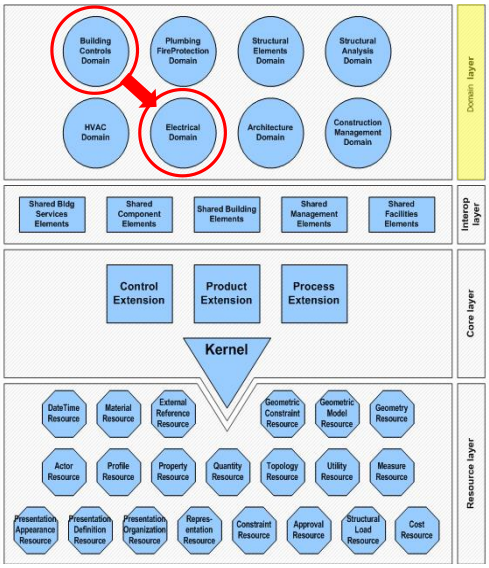


Figure 7. Domain Layer

Starting with Revit2023 version, the ifc mapping of entities in a project is simplified compared to previous versions. In fact, the program now natively contains the necessary parameters and, therefore, it is no longer essential to create new shared parameters "IfcExportAs" for IFC classes and "IfcExportType" for types and classify them as IFC parameters.

3.2 BACnet interface in BMS

EcoStruxure is an architecture and platform created and promoted by Schneider Electric, enabled for IoT technologies. EcoStruxure Building Operation (EBO) is a building management software that merges systems and application data to simplify the monitoring, administration, and optimization of building executive life.

In EBO, a first step is to create a BACnet Interface (Figure 8). From this point on, the basic BACnet settings are set: Instance ID, BACnet Name, Network ID. Instance ID is the value with which the Server communicates on BACnet; BACnet Name is the name of the device on BACnet; Network ID is equal to 1, is the standard BACnet IP network number and is editable (Figure 9).

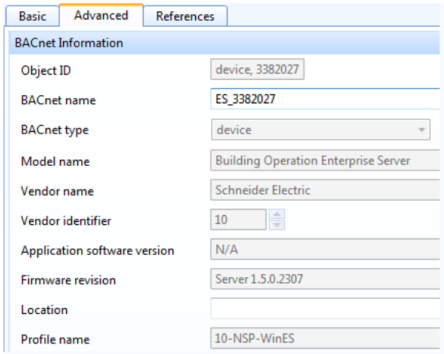


Figure 8. Creation of BACnet interface in EBO

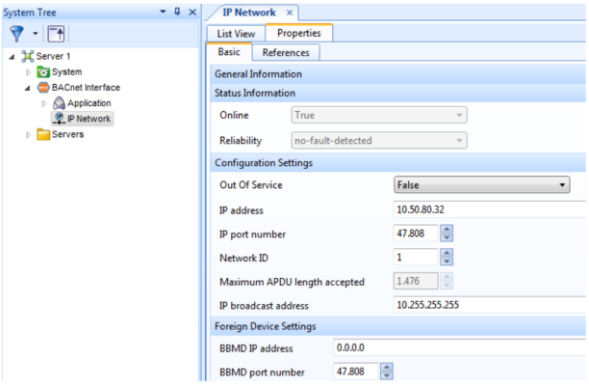


Figure 9. Editing BACnet IP Network

Figure 10 shows the scheme viewable in EBO.

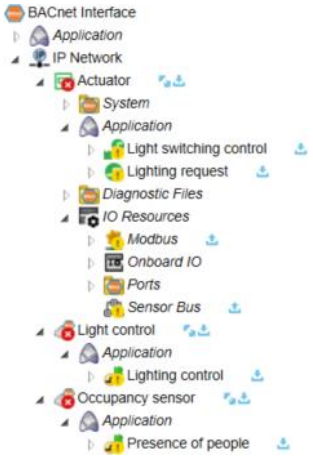


Figure 10. BACnet model in EBO

3.2.1 Export XML

The BMS model is exported in XML format. First, a BACnet interface was created containing the devices, populated with their objects. The structure is tree-like, exactly as it is presented within the EBO platform (Figure 11).

```
<ExportedObjects>
  <OI NAME="BACnet Interface" TYPE="bacnet.Device">
    <OI NAME="IP Network" TYPE="bacnet.IPDataLink">
      <PI Name="NetworkId" Value="1"/>
      <OI NAME="Actuator" TYPE="bacnet.mpx.udt.RPC12A">
        <OI NAME="IO Resources" TYPE="bacnet.mpx.IOResources" declared="1">
          <OI NAME="Ports" TYPE="bacnet.mpx.PortsFolder" declared="1">
            <OI DESCR="Room Bus" NAME="RS485-COMB" TYPE="bacnet.mpx.ports.RS485" declared="1"/>
            <OI DESCR="Sensor Bus" NAME="RS485-COMA" TYPE="bacnet.mpx.ports.RS485" declared="1"/>
            <OI DESCR="USB" NAME="USB" TYPE="bacnet.mpx.ports.USBPort" declared="1"/>
```

Figure 11. Actuator BACnet interface exported form BMS in XML file

3.2.2 Visual Studio Code for parsing IFC to XML

Having studied such a structure of the XML file exported by EBO, it is possible to create an algorithm through Visual Studio Code starting from the IFC file exported by Revit, to create a file with XML extension that can be imported into EBO and that would make automated the creation of a basic project like the one under analysis. This feature would be an essential discovery that simplifies and speeds up the process, without any loss of information. The work utilizes the IfcOpenshell library, to operate on the IFC file, and the ElementTree library to analyze XML file (Figure 11).

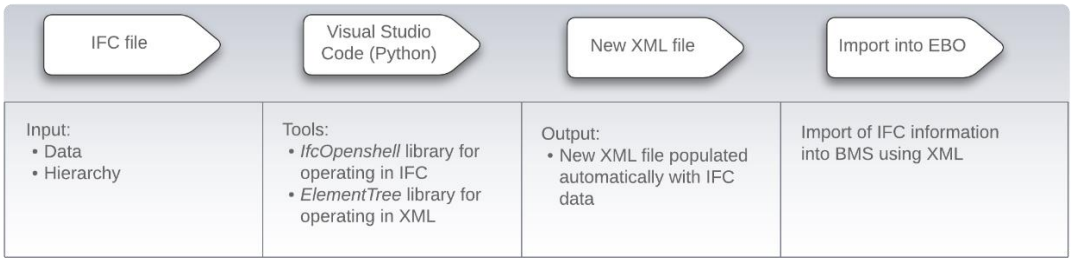


Figure 12. Mapping IFC integration with XML

Each device that exists within the EBO interface, and therefore in the exported XML file, is called by two common parameters, "NAME" and "TYPE" (Figure 13). The first one is the name associated by the user when creating the device, the second one is the template chosen from those proposed by EBO.

```
134          <OI NAME="Sensor" TYPE="bacnet.DeviceProxy">
```

Figure 13. NAME and TYPE for Sensor in XML file

Therefore, in Python, a new object is created and through the command "NewObj.set" the property "NAME" is assigned to it. Next, this parameter is set to the object.

It turns out that the original device data structure created with Python from the IFC file is the same as the official one exported from EBO. For a more accurate approach we need to align the type obtained from Revit with the one written in EBO, either by creating a new IFC parameter in Revit capable of returning the desired type or by generating a new family containing all the BACnet properties.

Listing 1. Set NAME and TYPE of Sensor using Python library for reading and writing IFC file information

```
Sensors=IFC.by_type("IfcSensor")
for sensor in Sensors:
    NewObj=ET.Element("OI")
    NewObj.set("NAME","Sensor")
    objtype= Element.get_type(sensor)
    typename=objtype.Name
```

```
52        NewObj.set("TYPE",typename)
```

```
10    <OI NAME="Sensor" TYPE="=" SE - SmartX - Optimum Housing - Touch Screen with Off-screen Control">
```

Figure 14. NAME and TYPE of Sensor from Python code in new XML file

In conclusion, it is necessary to populate the external application. This can be done through a feature that can return the XML entity with properties and values starting from its root. The same import procedure was also applied with the Niagara platform obtaining the same results.

4 Results

The experimental results demonstrate all the great potential that IFC has in serving as a common base for information exchange. The work focused on processing information output from the IFC database. Once the incoming information exchange to the database is also studied, it is possible to focus on defining the MVD to compile an exhaustive IDM.

The code, thanks to Python libraries, queries all the automation elements within an IFC-type project, groups them into a list and returns them in a readable file format that can be directly integrated within EcoStruxure building Operation software. Regardless of the complexity of the devices and objects, the code can read the elements information and write them into BACnet entities.

In this experimentation, secondary parameters descending from related parent entities, such as the digital and universal input and output ports of the considered actuator, were not implemented in the code because they are generated directly into the BMS application. Such parameters could be generated in Python and then printed to the XML file using computer language. This involved creating entities without them having a representation on the Revit model.

This work can offer a comprehensive and streamlined approach to writing foundational code in the design process of building automation and control systems. For instance, in scenarios like modeling BACS for a hospital or office building with thousands of devices to be programmed, having an available system architecture can significantly simplify and accelerate the design process.

5 Conclusion

This research has explored the dialogue between digital information systems and building control and automation systems, considering their increasingly strategic and fundamental role in the construction industry. Non-proprietary open BIM standards, such as IFC, serve a crucial role in modeling and exchanging information regarding automation. They enable consistent data storage and exchange among project team members, whether across platforms with similar or different software, thus unifying workflows. However, the literature review revealed that BACnet stands out as the sole standard enabling the modeling of BMS systems by selecting devices according to their standard profile and interoperability characteristics to be implemented. Hence, it facilitates modeling without requiring the declaration of specific actual devices. Through experimentation, it was demonstrated how data can be satisfactorily transferred from an IFC-format model to an XML-format file, automating the modeling of building automation systems in specialized software that adheres to the BACnet standard.

The code is based on the structure of the XML file exported from EBO, but the process can be standardized and automated to facilitate data communication and make the modeling of a BMS system even more immediate. It is envisaged in the future the possibility of studying the structure of other export files and other communication protocols of BMS platforms to create an interface with IFC as complete as possible.

The script that was generated was similarly replicated to import an XML file into Niagara. This demonstrates the potential that IFC can have in being a common base.

References

American Society of Heating Refrigerating and Air-Conditioning Engineers (2020). BACnet™, the ASHRAE Building Automation and Control Networking Protocol. URL https://www.ashrae.org/technical-resources/bookstore/bacnet (accessed 05.02.2024).

buildingSMART International (2020). Industry Foundation Classes 4.0.2.1 Version 4.0 - Addendum 2 - Technical Corrigendum 1. URL https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/ (accessed 05.02.2024).

buildingSMART International (2020). 7.4 IfcElectricalDomain. URL https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/ (accessed 04.30.2024).

buildingSMART International (2020). 7.2 IfcBuildingControlsDomain. URL https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/ (accessed 04.30.2024).

buildingSMART International (2024). Information Delivery Manual (IDM). URL https://technical.buildingsmart.org/standards/information-delivery-manual/ (accessed 05.02.2024).

buildingSMART International (2024). Model View Definition (MVD) - An Introduction. URL https://technical.buildingsmart.org/standards/ifc/mvd/ (accessed 05.02.2024).

Choudhary, Dr.S., Meena, G. (2022). Internet of Things: Protocols, Applications and Security Issues. Procedia Computer Science 215, 274–288 (https://doi.org/10.1016/j.procs.2022.12.030)

Dave, B., Buda, A., Nurminen, A., Främling, K. (2018). A framework for integrating BIM and IoT through open standards. Automation in Construction 95, 35–45. (https://doi.org/10.1016/j.autcon.2018.07.022)

Davis, D., Karlshøj, J., See, R., 2012. An Integrated Process for Delivering IFC Based Data Exchange.

Karavan, A., Neugebauer, M., Kabitzsch, K. (2005). INTEGRATION OF BUILDING AUTOMATION NETWORK DESIGN AND 3D CONSTRUCTION TOOLS BY IFC STANDARD. IFAC Proceedings Volumes 38, 247–254 (https://doi.org/10.3182/20051114-2-MX-3901.00034)

Kastner, W., Neugschwandtner, G., Soucek, S., Newman, H.M. (2005). Communication systems for building automation and control. Proc. IEEE 93, 1178–1203. (https://doi.org/10.1109/JPROC.2005.849726)

Li, H., Liu, H., Liu, Y., Wang, Y. (2016). AN OBJECT-RELATIONAL IFC STORAGE MODEL BASED ON ORACLE DATABASE. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. XLI-B2, 625–631. (https://doi.org/10.5194/isprsarchives-XLI-B2-625-2016)

Lohia, K., Jain, Y., Patel, C., Doshi, N. (2019). Open Communication Protocols for Building Automation Systems. Procedia Computer Science 160, 723–727. (https://doi.org/10.1016/j.procs.2019.11.020)

Madakam, S., Ramaswamy, R., Tripathi, S. (2015). Internet of Things (IoT): A Literature Review. JCC 03, 164–173 (https://doi.org/10.4236/jcc.2015.35021)

McGibney, A., Rea, S., Ploennigs, J. (2016). Open BMS - IoT driven architecture for the internet of buildings, in: IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society. Presented at the IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, IEEE, Florence, Italy, pp. 7071–7076 (https://doi.org/10.1109/IECON.2016.7793635)

Tang, S., Shelden, D.R., Eastman, C.M., Pishdad-Bozorgi, P., Gao, X. (2020). BIM assisted Building Automation System information exchange using BACnet and IFC. Automation in Construction 110, 103049 (https://doi.org/10.1016/j.autcon.2019.103049)

Wang, S., Xie, J. (2002). Integrating Building Management System and facilities management on the Internet. Automation in Construction 11, 707–715 (https://doi.org/10.1016/S0926-5805(02)00011-0)

Wang, S., Xu, Z., Li, H., Hong, J., Shi, W. (2004). Investigation on intelligent building standard communication protocols and application of IT technologies. Automation in Construction 13, 607–619. (https://doi.org/10.1016/j.autcon.2004.04.008)