Bauhaus-Universität Weimar
20.-22. September 2022
BauSIM2022 in Weimar
ibpsa

INTEGRATED DESCRIPTION OF TECHNICAL BUILDING SERVICES IN BIM

Dominik Schlütter[1], Nicolas Pauen[1], Jérôme Frisch[1], Christoph van Treeck[1]
[1] *E3D - Institute of Energy Efficiency and Sustainable Building, RWTH Aachen University, Germany,*
E-Mail: schluetter@e3d.rwth-aachen.de

Abstract

The digitalization of the construction sector is a crucial instrument to reach our goals for sustainable resource utilisation and energy systems. Within the research project Energie.Digital, a systematic method for linking the domains of building services engineering and building operation is developed and integrated into the Building Information Modelling (BIM) framework.

To connect the existing data silos from different domains, a hybrid approach to ensure interoperation was chosen. While linked data methods make it feasible to add semantic enrichment to this data, flat databases can also be attached. By implementing an extensible software stack that provides a lightweight API, various kinds of data sources can be addressed and interconnected while still using their own persistent backends.

Introduction

Building Information Modelling (BIM) is getting established in the field of Architecture, Engineering, Construction and Operations (AECO) to implement interoperability between different domains (van Treeck, 2016). Industry Foundation Classes (IFC) (ISO 16739, 2018) provide an open, standardized format for information exchange for a wide variety of concepts used in the AECO industries.

Currently, the focus of BIM interoperability is set on geometric information with coordination and clash detection as a strong driver to create a collision free coordination model. Product information properties are used within the models to support quantity surveying and tendering (Borrmann, et al., 2021). Furthermore, the IFC data model supports the use of functional structures und topological information which are not fully supported by the AECO software industry yet. Various attempts were made to explore these possibilities within the realm of IFC, but usually they involve intricate knowledge of the IFC standard and meticulous detail work while modelling the systems (Benndorf, et al., 2017).

Especially within building services engineering and building automation, a lot of information such as schematic diagrams, state diagrams and automation schemes are only provided in proprietary formats or as PDF representations (Schneider, et al., 2017). This effectively hides this information from traditional BIM viewer applications.

To mitigate these effects, a domain driven approach should be used to collect the information where it is generated. A plug-in-based software stack to add this domain specific knowledge together with a BIM model into a combined data source is presented in this paper.

The paper is structured into five parts: following a short overview of related work, the general software structure, and the key concepts for aligning the different data sources in the backend application are outlined. In the second part, the current plug-ins for the backend are presented, together with a description of frontend applications and a short digression on API structure. The third part shows an example of the plug-in mechanism, followed by a short discussion on results, and the last part concludes the paper with closing remarks.

Related Work

Some existing approaches already try to improve interoperability with a combination of different models. In (Rasmussen, et al., 2017) a system using the commercial Autodesk Forge Toolkit is described, this version is tightly integrated with the Revit BIM authoring software. The internal Revit GUID is used to connect the elements with a linked data system based on RDF and the BOT ontology (Rasmussen, et

al., 2021). The focus with this approach is set on architectural topology, the realm of building services systems is beyond the scope of this implementation.

Another established project is the Open Source BIMserver from bimserver.org (Beetz, et al., 2010). This provides a complete solution including support for revisions, user management, model checking, an API and a WebGL based viewer. It is used in various projects, especially in academia due to the focus on open source. Unfortunately, the proposed client integration with the standard authoring tools in the AECO industry is still lacking and a file based IFC workflow must be applied. It is a big project written in Java with more than 10 years of history and a rather steep learning curve. The Open Source BIMserver provides a model based IFC workflow, it can be seen as an extended IFC database to help querying, merging, and filtering IFC building data.

ConnectorHub

Within the project Energie.Digital a hybrid approach to handle BIM models was developed: Semantic enrichment using linked data has been a method to generate a topological view on building services with the TUBES System Ontology (Pauen, et al., 2021) and was also used to create an integral function description of building services (Ihlenburg, et al., 2020). On the other hand, flat data like time series and information from the building automation also needed to be integrated into the model.

With the aim for a lightweight and extensible framework, the main components of the software stack consist of a backend server, an API layer and frontend software to display results (see Figure 1). Both the backend and the frontend allow for a modular approach, so that different plugins or viewers can be adapted to fit the project structure.

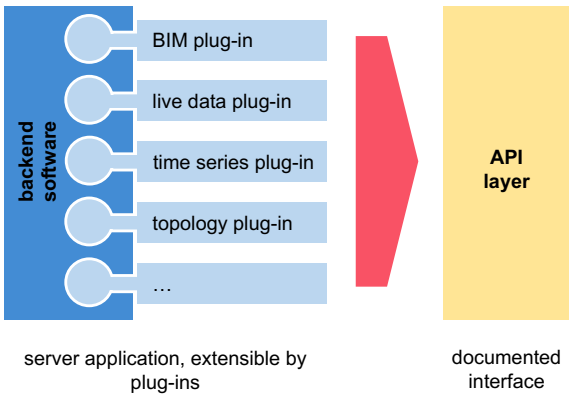

Figure 1: general backend software architecture

Systems Architecture

The backend server creates a thin abstraction layer to forward API calls to different connectors or plug-ins, which can provide an in-depth analysis on individual aspects of the BIM model. These connectors also implement or supply their own data storage for the results of the initial processing steps on the model.

There is a common file storage to share the IFC input files, as well as processed geometry (e.g., in GLTF format). It is also used for the generated static data files for the frontend viewer applications. For interactive use, a Redis Cache provides metadata via the BIM plug-in, this can be queried from the other plug-ins as well. The backend server is implemented using Python and FastAPI, which makes it easy to provide an OpenAPI compatible description as well as serving as an interactive documentation and test bed for the API layer.

Referencing and Alignment

One of the first design challenges was to identify a common key to link the various data silos. Initially, the reference designation key was chosen, as it is tied to the internal structure of the building services systems. It also won't change over time with different revisions from the BIM authoring software. This is not the case with the IFC GUID, which is only guaranteed to be unique in the current model. Some BIM authoring tools try to preserve IFC GUID after an initial export, but this is not mandatory. Unfortunately, a lot of subordinate components like pipes, ducts or fittings will not get their own reference designation key. It is still important to keep it and to be able to link elements with it, as it is an important anchor to compare revisions and it will be used during commissioning and operation of the building, and we want to be able to incorporate the complete building life cycle.

As the main input for our system will be IFC files, we settled with the IFC GUID to connect information from the BIM model, the topology graph, and the visualization in the frontend apps. This might require some effort when dealing with revisions, but on the other hand it nicely ties into the main visualization frameworks, which also use the IFC GUID to identify elements.

The backend server must provide a mapping for these keys. It is also necessary to be able to get information on an element using any of the keys available, so these queries:

```
GET /<plugin>/<data>?ifc_guid=1sz7SlB2j11[…]
GET /<plugin>/<data>?aks=DE2020_0601011_42[…]
GET /<plugin>/<data>?uri=https%3A%2F%2F[…]
```

should all generate the same response, as they all identify the same building element that was generated from the initial IFC model. This can be implemented using naming conventions, with IFC properties or with an explicit mapping table.

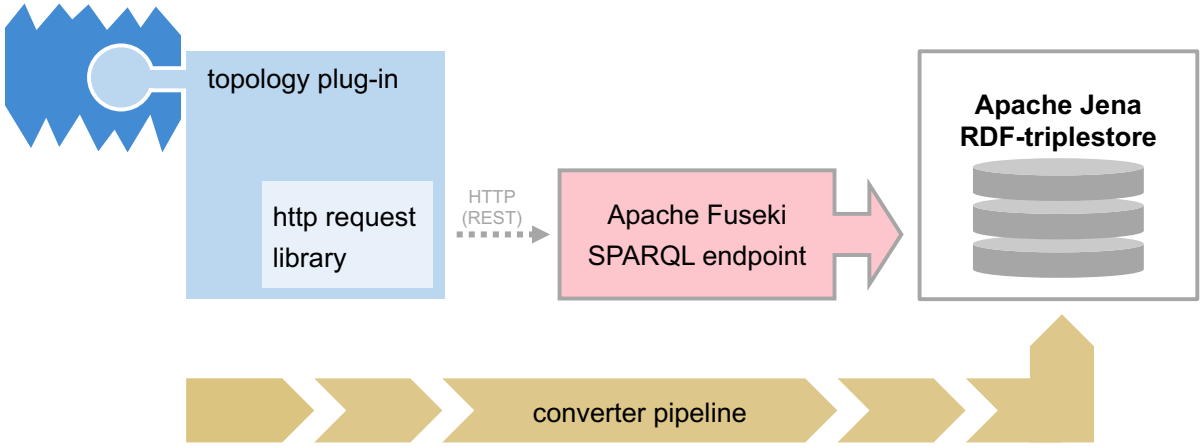

Figure 2: data flow topology plug-in

Connectors: backend plug-ins

While the backend server itself tries to provide an abstraction layer to combine the different data silos, each of them is accessible from within a backend connector plug-in.

The main starting point for alphanumeric metadata from the model is the BIM plug-in, a connector to provide information from the properties of IFC elements. During the initial import, those are extracted in a converter pipeline using ifcopenshell and then stored in a Redis cache for easy access. Some additional relations are also extracted: e.g., the parent element or the *IfcSystem* the element is part of. These are mainly used for filtering in the frontend applications, but they can also serve as validation targets for the topological analysis.

Another plug-in would be the topology connector to generate topological data of the building services systems as a directed graph in Resource Description Framework (RDF) format using the TUBES System Ontology (TSO) (Pauen, et al., 2020). With this semantic enrichment, it is possible, for example, to follow the systems graph from source to sink or to ensure the sequence of devices in its path. An Apache Jena Fuseki triple store and SPARQL server for graph storage is used as persistence layer. The graph is initially generated from an analysis of the IFC file, which is generated with a custom converter pipeline triggered by the file upload (see Figure 2).

Contemporary building automation generates lots of data points from building services. This information can be linked to components and presented as additional metadata. The live data plug-in provides this link by connecting with the building control system. A similar connection can be used to get monitoring or time series data, although the data source might be different and could even be an external service provider. This will be demonstrated in the Application example below.

The general idea for these plug-ins is their connection to a persistent data store that can be filled either by an adapted converter pipeline that operates on some kind of input files (e.g., IFC), or it can be an external data source. These need some conventions on the addressing, as they don't always use the reference designation key but chose their own addressing scheme.

Protocols

While plug-ins can be developed to integrate additional metadata of all kinds, some basic conventions have been enforced. To display content in the metadata info window of the default web frontend, the plugin needs to provide an endpoint for an "info"-route:

```
GET /<plugin>/info/{p_id}/{m_id}?ifc_guid=…
```

or

```
GET /<plugin>/info/{p_id}/{m_id}?aks=…
```

respectively, with *p_id* standing for the project identifier and *m_id* for the model identifier. This should return a JSON object with the attributes for this specific element for this connector plug-in.

As there might be multiple plug-ins providing the same kind of data, they also need to create the same JSON structure. This enables the frontend application to display data regardless of the originating source.

Frontend Applications

As the aggregated BIM data is available via static files and the REST API from the backend server, the addition of viewers and other frontend applications is rather straightforward. This enables the creation of different new views on the building data. Besides the Web-Frontend mentioned here, there might also be an

augmented reality (AR) application for mobile devices to explore the building services systems on location.

The standard web frontend is based on the xeokit framework, which offers various BIM related features. In addition to that, a compact management interface to select and upload project IFC files and other file-based data was implemented. The integrated BIM viewer had to be extended to enable communication with our backend server and the main menu structure needed to be adapted. The selection menu is organized following the DIN EN ISO 81346 (DIN EN ISO 81246, 2010) for reference designations using Function, Products and Location in addition to a models tab to select and enable different model files in the project. From this selection menu the view may be restricted by the hierarchy of elements shown (see Figure 3).

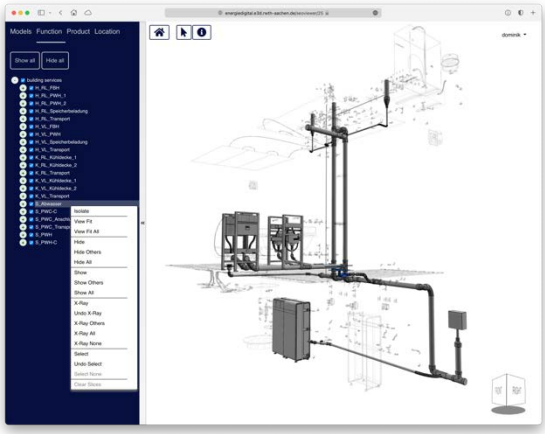

Figure 3: web frontend with selection view for the wastewater system

Using a visual selection of elements, a metadata inspector window allows to view aggregated data for this element, organized by plug-in. This would show the BIM metadata from the IFC file using information from the common property sets like Environmental Impact Indicators or Manufacturer Type Information as well as custom property sets inserted by the authoring software manufacturer or the contractor.

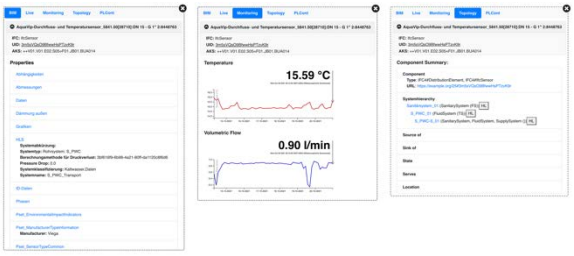

Figure 4: different tabs from the metadata information window

The metadata info window serves as an outlet for the different connectors to display a general overview regarding to this particular type of metadata (see Figure 4). As it is tied to the context of the component, it is possible to start a break-out view with detailed information without the confines of the BIM viewer web application.

Figure 5: web frontend with live data

If the building is already commissioned or in operation, a live data tab shows the current values from the building control system available for this component (see Figure 5). A time series connector provides additional data on those values over longer periods, showing curves for a quick analysis.

The topology tab gives an overview of the general connections of the element with regard to the building and the building services system it is placed in. For the latter, this means a link to its topological neighbours within the system (or adjacent systems). It provides information on the source and sink elements as well as the architectural entities it provides a service for. It can be used as a starting point for further topological analysis, and it can also be a source for other plugins to get access to topological information – as the live data plugin might show the surrounding sensors and values upstream or downstream.

Application Example

A potable water testbed was chosen to implement the concepts for the project. A very detailed as-built model provided a starting point to be imported into the software stack. According to the scientific background of this installation, multiple sensors were available, not only integrated into the active components but also along the piping.

Using the plug-in-based approach, the first version of the live data plug-in did query an InfluxDB with time series data that was generated by a script based on target values. When the building automation gateway was available, this was switched over – so now the actual values could be read with minimal delay from the database (see Figure 6).

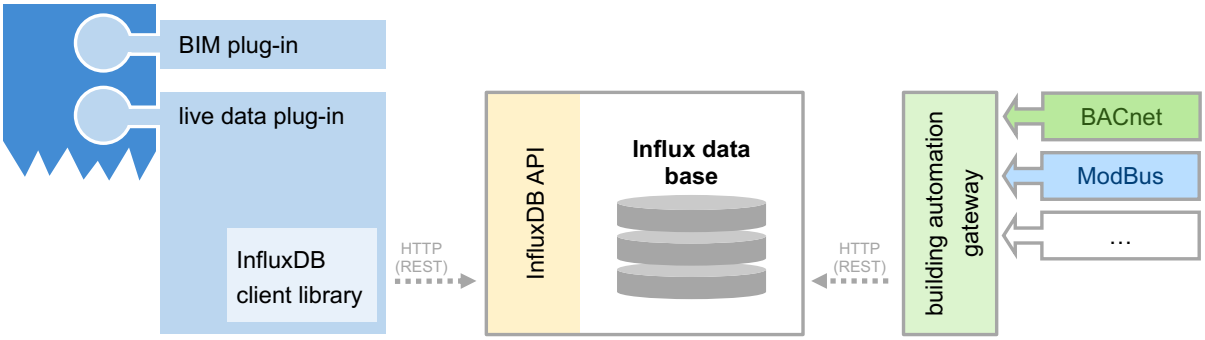

Figure 6: live data flow using InfluxDB

It is also possible to integrate other information into the frontend application – within the Energie.Digital project we have the PLCont connector to show data about the current state and functions of the building automation system for selected elements.

When a new type of building automation gateway was available, this structure made it feasible to write another plug-in querying the REST API of this gateway directly.

The findings from the Energie.Digital project will be validated on a larger scale building currently under construction. To get live data there, yet another building automation gateway had to be queried. During a pilot run it became obvious, that this gateway was rather limited, as it was only able to provide the last value from building automation for a given sensor id without any kind of metadata like the name or type of the sensor, or the SI unit used.

But all the information was available from the scientific monitoring system. Unfortunately, this data was saved in batches and might be several minutes old already. So, a new plugin was written, that first queries the monitoring system by reference designation key. This returns all metadata as well as a small timeseries and the sensor id, which could then be used to get another low latency datapoint directly from the building management system (see Figure 7). The results are combined into one JSON object, that can be consumed by the frontend application.

As all data gets normalized within the plug-in that is adapted to the currently used building automation gateway, so the resulting JSON object is independent from the gateway used.

```
"metadata": {

    "type": "temperature",
    "unit": "°C",
    "color": "#cc0000"

},

"values": [

    ["2022-03-11T18:02:00Z", 69.117],
    ["2022-03-11T18:03:00Z", 69.084],
    ["2022-03-11T18:04:00Z", 69.352],
    ["2022-03-11T18:05:00Z", 69.42],
    […]

]
```

The frontend applications did not have to change when the implementation of the backend connector was fixed.

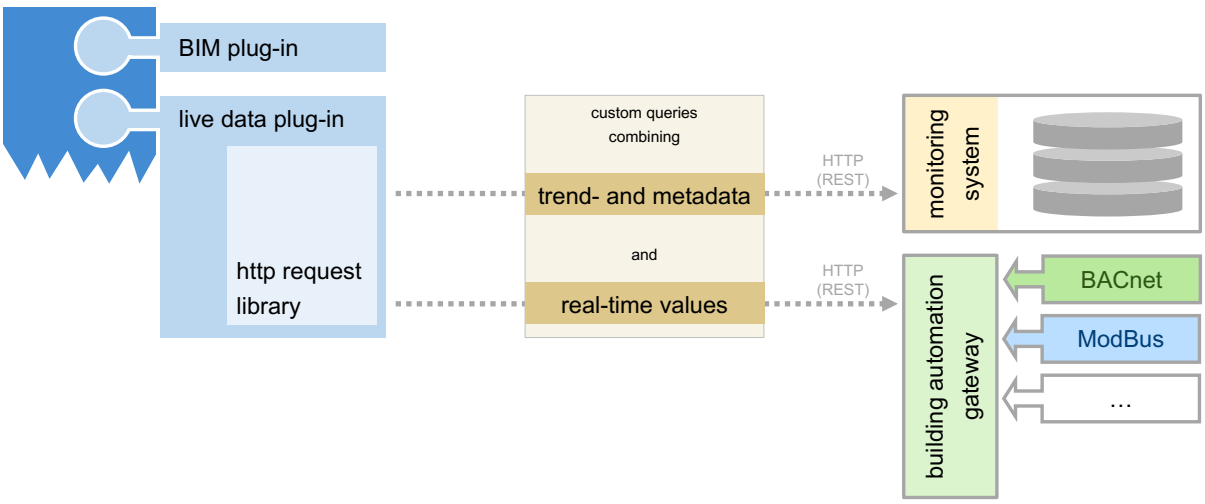

Figure 7: live data flow using combined sources

Results

The system is currently used as a prototypical solution within the confines of a research project for a small testbed installation. It was implemented to separate the processing of BIM data from the visualisation for the end user and using loose coupling for the domain specific plug-ins did facilitate the development process.

As the requirements changed within the testbed installation, we were able to adapt the data acquisition plug-ins to the new sources without impacts on the viewer part. On the other hand, new features and API endpoints can be created in the domain specific part without touching the main structure or other plug-ins.

The minimal requirements for plug-ins make it easy to develop something, but the lack of a bigger ecosystem and clients to consume the API endpoints lead to rather minimalistic interfaces.

While the current focus is set on support for domain independent visualisation tools, the ConnectorHub framework would benefit from some extended set of API endpoints that are common to all plug-ins.

And although the software stack was created with extensibility in mind, it has been used with a very limited set of building data yet.

Conclusion and Outlook

The basic approach using an extendable, plugin-based infrastructure to combine specific domain knowledge with the geometry and metadata from BIM models seems to be a plausible way to get an integrated description of technical building services. The current prototype will need to be refined, especially regarding the structure of the API endpoints – providing more than just one basic information response.

Within the project Energie.Digital the various methods to integrate the different data silos will be evaluated during construction, commissioning and operation of a seminar centre which is scheduled to open at the end of 2022. This will be an opportunity to test the software framework with a broader range of BIM data.

Acknowledgements

The research within the project Energie.Digital leading to these results has received funding from the German Ministry for Industry and Energy under grant agreement no. 03ET1611. The authors would like to thank the Fraunhofer Institute for Solar Energy Systems ISE and Viega GmbH & Co. KG for their contribution to the project.

References

Beetz, J., van Berlo, L., de Laat, R., van den Helm, P. 2010. Bimserver.org - an Open Source IFC model server. Rotterdam, International Council for Research and Innovation in Building and Construction (CIB).

Benndorf, G., Réhault, N., Clairembault, M., Rist, T. 2017. Describing HVAC controls in IFC – Method and application. Energy Procedia, Volume 122.

Borrmann, A., König, M., Koch, C., Beetz, J. 2021. Building Information Modeling - Technologische Grundlagen und industrielle Praxis, Berlin, Springer Vieweg.

DIN EN 81346-1:2010-05, Industrielle Systeme, Anlagen und Ausrüstungen und Industrieprodukte – Strukturierungsprinzipien und Referenzkennzeichnung – Teil 1: Allgemeine Regeln (IEC 81346-1:2009).

Ihlenburg, M., Rist, T., Benndorf, G., Réhault, N. 2020. A hybrid method for an integral function description of building services, Proceedings of BauSIM 2020 - 8th Conference of IBPSA Germany and Austria.

ISO 16739-1:2018: Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries – Part 1: Data schema.

Pauen, N., Schlütter, D., Siwiecki, J., Frisch, J., van Treeck, C. 2020. Integrated representation of building service systems: topology extraction and TUBES ontology. Bauphysik 42 (6).

Pauen, N., Schlütter, D., Frisch, J., van Treeck, C. 2021. TUBES System Ontology: Digitalization of building service systems Proceedings of the 9th Linked Data in Architecture and Construction Workshop.

Rasmussen, M. H., Hviid, C. A. & Karlshøj, J. 2017. Web-based topology queries on a BIM model.

Rasmussen, M. H., Lefrançois, M., Schneider, G. F., Pauwels, P. 2021. BOT: the Building Topology Ontology of the W3C Linked Building Data Group. Semantic Web Journal, Volume 12.

Schneider, G. F., Pauwels, P., Steiger, S. 2017. Ontology-based modeling of control logic in building automation systems. IEEE Transactions on Industrial Informatics, Volume 13.

van Treeck, C. 2016. Building Information Modeling. in Gebäude. Energie. Digital. Berlin, Springer Vieweg