Contesting the lethal Mediterranean frontier

*Published Version:*
Heller, C., Pezzani, L., Stierl, M. (2023). Contesting the lethal Mediterranean frontier. Cheltenham : Edward Elgar Publishing [10.4337/9781800377509.00043].

*Published:*
DOI: http://doi.org/10.4337/9781800377509.00043

# Digital Decision Support System Prototyping for Building Performance Analysis and Management

Angelo Massafra[1][0000-0002-3642-4660], Ugo Maria Coraglia[1][0000-0001-7781-887X] , Giorgia Predari[1][0000-0001-7422-1744] and Riccardo Gulli[1][0000-0001-5720-2912]

[1] Alma Mater Studiorum University of Bologna, Bologna, Italy
`angelo.massafra2@unibo.it`

**Abstract.** The ongoing transformation of the AECO sector towards digitalization has led to a growing need for digital decision support systems (DDSS) to aid in managing built heritage. While there have been many technological improvements in this area, creating these digital tools still demands substantial technical and financial investments and highly specialized IT competencies. To respond to this challenge, this paper presents BTwin, a toolkit developed to facilitate the prototyping processes of DDSSs for performance-oriented building management. This open-source software, implemented in Python, allows for integrating building data from multiple sources into graph networks, such as building information models and building performance simulations, meters, and sensors. This integration capability, supported by specific semantic and ontological rules, is complemented by the possibility of quickly displaying the data on interactive dashboards accessible to non-expert users. After explaining the theoretical framework behind the toolkit, the paper showcases its practical application in a university building, focusing on energy- and occupancy-related topics.

**Keywords:** Built Heritage, Decision Support Systems, Knowledge Graph, Digital Twin, Building Information Modelling.

## 1 Introduction

The built environment is responsible for significant global energy consumption and resultant emissions [1]. Despite it is at the crossroads of many EU policies aimed at operating buildings more sustainably [2], its digital and ecological transformation is hindered by the recognized limited innovation that characterizes the Architecture, Engineering, Construction, and Operations (AECO) sector [3]. To address the innovation challenges in the field, the push for innovative technologies such as Building Information Modeling (BIM), the Internet of Things (IoT), Building Performance Simulation (BPS), Artificial Intelligence (AI), and, more recently, Digital Twins (DT), has stemmed in recent years [4] for better decision-making in built asset management [5].

While DT technology has undergone rapid development, it is still in its early stages, needing both prototyping and technical development, as well as more conceptualization [6]. The lack of standardized definitions, coupled with the heterogeneous and diverse range of data and data sources and the unscientific approach typically associated with building management [7], generate an obstacle to the organic growth of this technology and, more generally, to that of Digital Decision Support Systems (DDSS) for buildings, limiting the practical utilization of the many data (that already characterizes building management practices) as practical knowledge.

Nowadays, these systems require significant resources for development, with cultural, financial, and economic benefits that can be sometimes difficult to grasp during the conceptualization phase. Moreover, to enable secure and simplified data flows, as well as the creation of reliable, scientifically valid, and non-digital-expert accessible knowledge, these digital solutions require high expertise in many fields, including computer science, electronics, construction science, building physics, and data science. The scientific community is thus primarily called upon to theorize such systems and their semantics, as, at the small scale, it may lack the resources or technical personnel sufficient for the development of these systems. Secondly, it is asked to prototype solutions that demonstrate the concrete benefits of these technologies to building managers, who should be considered their main beneficiaries.

The need to address both these requirements has led to the inception and development of BTwin, a software toolkit designed for simple, flexible, modular, and cost-effective prototyping of DDSS for building management. To achieve the objective of structuring and representing building knowledge, the functions within the toolkit allow for the semantic linking of building data (obtained from various sources, e.g., BPS, BIM, sensors, and meters) within graph networks, and then for its visualization through thematic interactive dashboards. Through low-code solutions, the toolkit allows prototyping of digital systems that can perform functions useful for building management focused on performance. For example, space usage monitoring and planning, analysis of energy needs (also in relation to space occupancy), reporting of energy and water consumption (parameterized on KPIs), estimation of emissions due to building usage, as well as analysis of indoor environmental quality regarding thermal, visual, acoustic comfort, and air exchange.

At the core of the software, developed in Python, lies a JSON (JavaScript Object Notation) data model, which is lightweight, quick to write, and human readable. This model was developed following the principles of object-oriented programming, with the main purpose of representing building data within graph structures following the axioms and meanings defined in a supporting ontology, whose development was made by federating some schemas of proven importance in the AECO sector. Specifically, the schemas considered for the toolkit include: the Industry Foundation Classes (IFC) for addressing spatial and construction-related aspects [8], the Building Topology Ontology (BOT) for building topology [9], Topologic's class hierarchy for spatial and topological aspects [10], Brick's ontology for system and sensing aspects [11], and the EnergyPlus' Input Data Format (IDF) for energy-related aspects.

The scalability and modularity of the tool's architecture are ensured by logical connectors. The toolkit is built upon many popular Python libraries, each serving a specific purpose, for example, Pandas for data handling [12], NetworkX for graph structuring and analysis [13], Topologicpy for spatial analysis [14], Eppy for energy simulation [15], Plotly for data visualization, and Dash for dashboard development [16].

The article is organized as follows. Initially, it provides the research background. Then, it explains the theory behind the toolkit and presents its technical details. Thirdly, it showcases the practical application of the toolkit through a workflow example aimed at developing DDSS for analyzing the energy performance of the spaces of a pilot building selected among Italian higher education-built heritage.

## 2 Background

### 2.1 Modeling building knowledge through graphs

The interaction between people, especially in workplaces and educational settings, corresponds to the creation of knowledge structures, which sometimes can be represented as knowledge graphs (KG).

Creating a graph means identifying specific objects and labeling their relationships to explain and/or share a problem, a domain, or a concept, with the essential purpose of transferring knowledge. Such graphs can be created by humans or computers. In the latter case, they are composed of nodes (or vertices), used to represent entities and subjects, and edges (or links), used for their connections, i.e. predicates.

Both semantics and property descriptions can be attributed to the nodes and edges that make up graphs [17]. The organization of information in such a way allows, on the one hand, to assign an ontological meaning to each element that composes a knowledge domain, and, on the other hand, to use attributes to describe every element. These properties can be qualitative or quantitative, textual or numerical, but not also.

Hence, when information is encapsulated within a KG, it becomes possible to convey it as valuable knowledge via processes such as extraction, aggregation, and analysis, regardless of the complexity of the underlying cognitive structures. In a similar way, to simplify the comprehension and management of a knowledge domain, a KG can be divided into several smaller graphs, effectively organizing the domain into more manageable subdomains.

Beyond these capabilities, KGs are highly efficient in managing and integrating data from different environments or formats and in analyzing Big Data [18]. Coupled with the Semantic Web, they enable excellent ease of data extraction and accessibility on the web.

All these features have allowed KGs to play a crucial role in digital applications, including in the AECO sector [19].

## 2.2 Reference ontologies in the AECO sector

Grasping the significance of the terms employed in a KG is crucial for methodically structuring its encompassed knowledge and conveying it to external parties in a distinctly intelligible way. Typically, ontologies are used within a KG to provide a structured depiction of information. They represent formal, shared, and explicit articulation of a domain's conceptual framework, designed to guarantee that every piece of information in that domain is precisely defined, thereby making it usable to logical analysis.

In recent times, paralleling the surge of digital resources in the construction industry, numerous ontologies have been crafted to delineate the AECO realm and its subsets. A comprehensive examination of the literature concerning the predominant ontologies for crafting building applications - with a primary emphasis on energy matters - is offered by Pritoni et al. [20]. The subsequent section succinctly presents only the schemas that were utilized to construct the federated ontology that supports the toolkit detailed in the following paragraphs. These are: IFC, BOT, Topologic, Brick, and IDF.

FC stands for Industry Foundation Classes and is the international open BIM standard. Its development and maintenance are supported by buildingSMART International. It serves as an open specification for BIM data, facilitating the exchange and sharing of information among different stakeholders involved in building construction or facility management projects [8]. BOT is a simplified ontology put forward by W3C that focuses solely on the fundamental concepts of a building's topology. It covers both physical and spatial components of a building and the interconnections between them, aiming to provide a concise framework for representing topological aspects within buildings. [9]. Topologic is a software library dedicated to spatial analysis in buildings, developed through a collaboration between Cardiff University and University College London in the UK. It is based on a theoretical class hierarchy designed to categorize building topological concepts and depict them using simplified geometric forms [10]. Brick is an open-source schema that provides a standardized framework for the semantic description of physical, logical, and virtual assets in buildings, along with their interrelations [11]. It is particularly used in contexts where the IFC format falls short in fully detailing diverse components of buildings, especially in the fields of electrical and mechanical systems where clear geometric references might be absent [21]. IDF stands for Input Data Format, which is used by EnergyPlus, a building energy simulation tool developed by the US Department of Energy. This format is designed to detail the energy-related aspects and parameters of buildings necessary for conducting energy simulations.

# 3 Methods and tools

## 3.1 Ontology definition

The first methodological step for developing the toolkit involves identifying an ontology capable of providing a formal representation of the performance-related aspects of existing buildings. In the case of this study, this representation is obtained by federating the above-mentioned schemas (section 2.2), i.e. by combining them into a single framework consistent with the research objectives through the alignment of the concepts defined within them.

The federation process ensures, on the one hand, standardization of information and, on the other, interoperability with external information systems that are built on one or more of the reference ontologies.

Relying on the federated ontology, BTwin allows modelling building knowledge within KGs, with the primary assumption that a building can be represented as a graph composed of nodes (subjects) and connections (predicates). The nodes are semantically distinguished by identifying their belonging to specific classes and subclasses, defined in the ontology, while the connections between nodes are characterized through reference to specific types of relationships, encoded in the ontology too.

The classes used for the semantic characterization of the entities are:

- *Topologies*: spatial components, essentially existing as physical or abstract constituents within the real or virtual world and identified by their positioning within a 3D space (e.g., spaces and storeys);
- *Interfaces*: abstract representations of partition elements bounding and linking topologies (e.g. walls, roofs and doors);
- *Equipment*: devices and systems serving the entirety or specific areas of a building (e.g., HVAC system and its devices, or a sensing network and its IoT devices).

As anticipated, these definitions are aligned with those of the ontologies presented in section 2. In particular, Topologies are described as 'IfcSpatialStructureElement' in IFC, 'Topology' in Topologic, 'Location' in Brick, and 'Zones' in BOT and IDF. Interfaces are represented as 'IfcElement' in IFC, 'Interface' in BOT, 'Topology' in Topologic, and 'Surface' in IDF. The Equipment class recalls, instead, the 'Equipment' class as described in Brick.

Each of the node classes has its subclasses.

Topologies can be distinguished into 'District', 'Site', 'Building', 'BuildingStorey', 'Zone', and 'Space' elements. These elements the spatial hierarchy of BTwin, which draws upon the spatial structure of IFC, enriched by the introduction of the 'District' class, defined as a geographical area or region within a larger administrative, political, or functional context, particularly useful in building management.

Interfaces are categorized into 'Face' and 'Aperture' elements, mirroring the classification found in Topologic. Faces encompass vertical, horizontal, or sloped boundaries that separate spaces and zones from other internal or external areas, including walls, roofs, and floors. 'Apertures' refer to openings within these faces that allow the flow of people, air, or light, and encompass elements like doors, windows, and holes.

Equipment elements are organized according to the equipment hierarchy, which encompasses 'System', 'Subsystem', and 'Device' subclasses. Systems are defined as aggregates of interlinked systems that fulfill particular functions across one or more buildings, such as a building's HVAC system. Subsystems refer to assemblies of interrelated components, which include equipment devices and infrastructure, devised to execute specific tasks within one or more zones, like a zone's heating system. Devices represent specialized components within a subsystem, tasked with specific functions and interconnected with other devices to constitute integrated systems, examples of which include radiators, pumps, or boilers.

Based on their belonging to the mentioned classes and subclasses, nodes can be related by various types of connections. Below are the main types of relationships supported by BTwin.

The 'HasLocation' and 'IsLocationOf' relationships determine whether an element is spatially contained within another or, inversely, if it encloses another element. These relationships can be utilized across various topologies (for instance, a building storey containing a space), between interfaces (such as a wall containing a window), or between equipment devices and topologies (like a technical room containing a chiller).

Interfaces and Topologies are linked via the 'IsInterface' and 'HasInterface' relationships. These connections primarily represent the adjacency between a Face (or Aperture) and the spatial elements it borders. They are instrumental in recognizing adjacency or passage between two spaces, particularly in simulations. For instance, when a partition wall acts as the interface between two distinct spaces, it indicates their adjacency. Likewise, when a door functions as the interface between two separate spaces, it denotes the presence of a passage connecting them.

Two additional relationships define the connections between Equipment elements and between Equipment and Topologies.

- The 'HasPart' and 'IsPartOf' relationships allow navigation through the equipment hierarchy. For example, a radiator is part of a heating system, which is itself part of a building's overall HVAC system.
- Conversely, the 'Feeds' and 'IsFedBy' relationship establishes connections between systems and subsystems with Topologies (such as a heating system serving a specific zone in a building) or between one device and another (like a boiler supplying heat to a radiator within a particular space).

Fig. 1 diagrammatically illustrates a summary of the classes, subclasses, and relationships that constitute the federated ontology of BTwin.

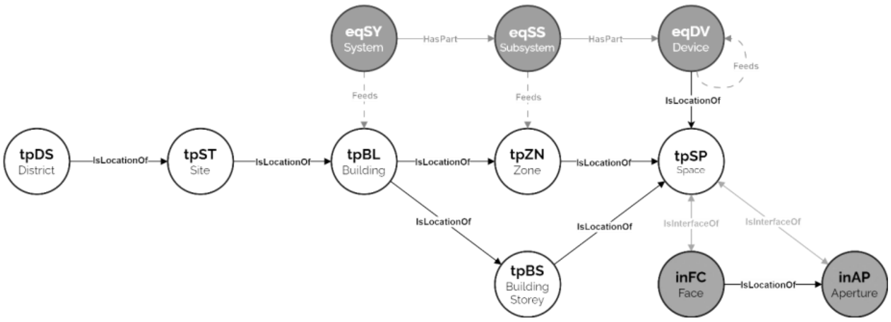

**Fig. 1.** Classes, subclasses, and relationships constituting the federated ontology supporting the BTwin toolkit. © 2023, A. Massafra

### 3.2 Attribute modeling

In addition to the semantic and hierarchical classification of nodes, the entities modeled within BTwin can be enriched with data and metadata that qualitatively or quantitatively describe the entities themselves. Specifically, such data can be attributed to the nodes as:

- *Points* signify isolated units of information detailing observations made at a specific time, effectively denoting the value tied to a device, system, or spatial entity devised for detecting and quantifying a variable (e.g. a temperature record from a sensor at a certain hour of a certain day);
- *PropertySets* (PSets) pertain to clusters of interconnected properties that are organized according to themes and types (e.g., the thermal set-points of a space describing its thermal requirements);
- *KPISets* (KPISets) encompass arrays of metrics, or key performance indicators (KPIs), adopted to evaluate the performance of spatial constituents within a building during a specified timeframe (e.g. thermal comfort-related metrics describing the indoor air quality of a space or energy-related metrics describing the energy needs of a zone).

Within BTwin's KGs, Points, PSets, and KPISets serve as nodes that can establish connections with other entities via specific relationships. When describing data derived from energy simulations, the 'IsPointOf' and 'HasPoint' relationships enable the attachment of Points, for instance, to Topologies like zones. Alternatively, Points can be directly attached to devices when they represent, for example, sensor measurements.

PSets can be linked with any type of entity through the 'IsPSetOf' and 'HasPSet' relationships. Instead, KPISets can connect to spatial elements through the 'IsKPISetOf' and 'HasKPISet' relationships. PSets and KPISets are respectively connected with Properties and KPIs through the 'HasProperties' and 'HasKPIs' relationships.

### 3.3 Data model syntax

Data modeling in BTwin is supported by a data model that relies on a particular syntax, which uses JSON objects to represent entities and their relationships. An advantage of this notation is that it makes the data easily writable and readable by humans - as well as by machines - especially when compared to formats like the EXPRESS schema used by IFC. Due to its extensibility and adaptability, JSON allows for the serialization of data within dictionaries, which can be interrelated to create graph structures. Additionally, this format is particularly well-suited for data exchange in web environments.

The fundamental assumption for representing information within the data model is that every building element - whether it is a Topology, an Interface, or an Equipment – must correspond to a JSON object equipped with its own unique identifier (UID). This JSON object takes the form of a dictionary, equipped at least with its own UID, the unique identifier of the subclass it belongs to, and possibly, indications of this instance object's relationships with other instances. All this information is included in the dictionary within key and value pairs.

Below is a syntactic description of some entities that can be categorized using the previously illustrated classes to clarify how the data modelling process occurs. Tab. 1 illustrates, on the left side, how an instance of Topology is represented within BTwin's data model. Within the JSON object, in the table the key 'UID' holds the unique identifier for the specified space, <cell45>, while the identifier 'tpSP', representing the space's subclass, is assigned to the 'Class' key. The space's relationships with other objects are defined under the 'relationships' key. The value corresponding to this key is another dictionary, where each type of relationship serves as a key, and the UID of the related object serves as the value. For instance, the placement of <cell45> within the storey <BuildingStorey_926cm> and within the zone <ZN_Circulation_A> is established by the key 'HasLocation'.

At the center, Tab. 1 displays instead the formatting of a Point according to the JSON-based syntax. The key 'UID' holds the unique identifier of the point, <pointX>, while the class identifier 'ptSM', assigned to the 'Class' key, represents the subclass of the point (where 'pt' stands for 'Point' and 'SM' for 'simulation'). The relationships between the point and other instances are defined under the 'relationships' key. Specifically, <pointX> is related to <ZN_Circulation_A>, which is a energy zone. This relationship is established by the key 'IsPointOf.' Moreover, this JSON object includes two additional keys. The key 'timestamp' refers to the time at which the value is simulated, expressed in the format YYMMDDhhmm. The key 'value', on the other hand, reports the value and the units of the measured parameter. In the example shown, a value of 0.15 kWh heating energy demand is simulated for <ZN_Circulation_A> at 10:00 on November 15, 2022.

Lastly, on the right side, Tab. 1 presents an example of a KPISet in JSON format. This KPISet assesses the performance of the building <buildingX> in terms of energy need. Within this JSON object, the key 'UID' contains the unique identifier, <OperationalEnergy>, for the KPISet in question. The relationships of the KPISet are outlined under the 'relationships' key. Essentially, the 'IsKPISetOf' relationship clarifies that <OperationalEnergy> is a KPISet for the building <buildingX>. The key ‘hasKPIs’ refers to the relationships of the KPISet with the indicators that comprise it, ‘kp_HeatingEnergyDemand’ and ‘kp_CoolingEnergyDemand.’ Lastly, the ‘timestamp’ key provides an indication of the period to which the KPISet pertains.

The notation of PSets is very similar to that of KPISets, with the difference that PSets do not have a ‘timestamp’ key and the ‘hasKPIs’ key is replaced by the ‘hasProperties’ key.

**Table 1.** JSON objects representing a Space, a Point and a KPISet according to the data model syntax

Topology

```
{
 "Class": "tpSP",
 "UID": "cell45",
 "relationships": {
   "HasLocation":[
     "ZN_Circulation_A",
     "BuildingStorey_926cm"]
}}
```

Point

```
{
 "Class": "ptSM",
 "UID": "pointX",
 "timestamp": "202211151000",
 "relationships": {
   "IsPointOf":["ZN_Circulation_A"]
 },

      "value":{

        "quantity":"HeatingEnergy_kWh",

        "value":0.15
                    }}
```

KPISet

```
{
 "UID": "OperationalEnergy",
 "Class": "ksKS",
 "relationships": {
   "IsKPISetOf": ["buildingX"]
 },
 "hasKPIs": {
   "k_HeatingEnergyDemand": {
     "value": 18.85,
     "quantity": "Energy_kWh"
   },
   "k_CoolingEnergyDemand": {
     "value": 5.44,
     "quantity":"Energy_kWh"
   },
 },
 "timestamp": {
   "from": 202201010100,
   "to": 202212312400
 }
}
```

### 3.4 Toolkit architecture

BTwin is a structured Python library supported by well-documented reference Python libraries whose implementation is widely supported by the AECO’s programming community.

As anticipated, this software capitalizes on the principles and syntax of JSON to facilitate rapid prototyping of dashboard based DDSSs. The JSON format employed can be interpreted and transformed into network graphs through specialized functions. Specifically, NetworkX is harnessed to build the graph database [13], a Python library equipped with an extensive array of tools designed for the creation, manipulation, and examination of the structure, dynamics, and functionalities of complex networks.

Employing what is referred to as 'connector' logic, the BTwin seamlessly incorporates external libraries and data formats. This design principle fosters scalability and modularity, significantly enhancing the development process of DDSS by allowing for flexible integration and expansion capabilities.

For instance, the connector integration with Topologicpy enables the utilization of models created in Topologic to represent spatial entities directly within the graphs. In a similar vein, by integrating the APIs of the Eppy library, the connector with EnergyPlus allows for the assimilation of energy simulation outcomes with the spatial elements being modeled for their energy behaviors. Other connectors provide a simplified means of interfacing with Autodesk Revit through Autodesk's APIs, which is instrumental in extracting spatial element properties from BIM models to link them with graph nodes. Additionally, there has been testing for compatibility with Arduino to model sensor data, and preliminary trials have been conducted with ChatGPT APIs to facilitate information extraction from graphs through textual prompts, further expanding the toolkit's versatility and application scope. The modular design of the library sets the stage for future expansions, including the development of connectors for more intricate data formats like IFC.

Once data are interlinked within the graph, whether through connectors or not, the toolkit offers a variety of functionalities for data querying, aggregation, processing, and visualization within the KGs. The visualization aspect, in particular, leverages the Plotly and Dash libraries to facilitate the creation of prototype web dashboards [16]. These can be transformed into web micro-applications, deployable on the Internet through platforms like Docker and AWS, a process exemplified by the work of Betti et al. [22].

## 4 Demonstration

### 4.1 Case study

In this section, the methodological application of the toolkit is presented, exemplifying a possible usage workflow. This workflow aims to develop a DDSS capable of informing building managers about the energy needs of various spaces and linking them to occupancy parameters using data calculated through dynamic energy simulations. Such calculations allow for high-resolution performance data, which, being aggregable and distributable on an hourly frequency, enables the examination of different seasonal conditions (e.g., winter/summer) and occupancy scenarios (e.g., weekdays/weekends). The outcome of the application is a dashboard-based interface that facilitates the assessment of electricity needs (for cooling, lighting, ventilation, and appliance use) and natural gas (for heating) of spaces, also estimating the costs and CO2 emissions into the atmosphere associated with these energy needs.

For demonstrative purposes, the theoretical framework previously presented is tested on a case study representative of the Italian building heritage managed by local public administrations, a large sample of buildings that forms about 10% of the entire national built heritage [23]. Specifically, the Faculty of Engineering of the University of Bologna is considered for demonstration, being a heritage building under the management of an Italian local public administration. Constructed from 1932 to 1935, it stands as one of the early 20th-century edifices to be recognized in the city, hailed as a landmark of local rationalist architectural heritage [24]. Spanning 19,200 square meters across four floors, it can accommodate up to 5,000 individuals, including researchers, staff, and students. On average, around 2,500 students frequent the building during academic hours, five days a week, for 11 months annually. A specific section of this building serves as the experimental area. This section accommodates the Department of Architecture (DA) and the Department of Civil, Chemical, Environmental, and Material Engineering (DICAM), comprising about 70 rooms across two floors with diverse uses such as offices, meeting rooms, libraries, common areas, and classrooms, representing the functional complexity typical of higher education facilities.

### 4.2 Workflow

The main steps followed for the DDSS implementation, carried out in the Jupyter Notebook programming environment using the BTwin Python library, are reported below.

Initially, the spatial hierarchy of the case study was created by modeling all the Topologies that compose the building within a KG. This step was automated by importing a BIM model generated semi-automatically through Topologic and Autodesk Revit.

Subsequently, the PSets related to spaces and zones relevant to the application were added to the KG, and relationships between these and the spatial elements were established. Specifically, information regarding dimensions (such as area, volume, and height) and occupancy conditions (maximum number of people and occupancy density) were added for each space. Some of these properties were revealed to be helpful in normalizing the KPIs in subsequent steps.

The data from the EnergyPlus simulations were formatted in JSON and modeled in the KG as Points, then linked to the zones that make up the building's energy model. These Points include parameters related to energy needs for the heating, cooling, and lighting of each zone within the building. For the simulations, an IDF energy model was used, which was created from the BIM model mentioned before using the procedure already described in [25].

All this information, assigned to the zones, was extracted from the KG to calculate, zone by zone, the energy needs KPIs. These KPIs, which could be aggregated with an hourly resolution, indicate the amount of electricity and heating required by the spaces, the related costs, the estimated CO2 emissions, as well as an indication of the internal gains from people (useful for understanding the occupancy patterns of various zones) and solar gains (useful for understanding how the exposure of spaces affects their energy behavior). For each space, these KPIs were grouped into KPISets and then linked to the zones in the KG.

Finally, using the Dash library, an interactive dashboard was created to display the KPIs on the BIM model's geometries and within charts and tables.

From a methodological standpoint, the process outlined below can be replicated for informational purposes different from that described here (e.g., energy consumption analysis, indoor air quality monitoring, thermal comfort assessment and evaluation of space usage conditions) and with different types of dynamic data (for instance, data from sensors could be used instead of simulation data). Future research activities will focus on testing the tool on more complex case studies and for various informational uses to assess its methodological and instrumental limits and potentialities.

### 4.3 Spatial hierarchy

The spatial hierarchy of the building was established in a BTwin's KG by reading the data within a BIM model developed through Topologic which is depicted in Fig. 2. This BIM model consists of spaces – defined in Topologic as 'Cells' – aggregated into a single model representing the building, known in Topologic as 'CellComplex'. In this model, each space is separated from another by 'Faces', which represent the building's partition elements. These include doors and windows, referred instead to as 'Apertures'. In the Topologic model, the properties of both Cells and Faces and Apertures are described using 'Dictionaries', which represent these properties within Python vocabularies through a system of key-value couples. Each of these elements has its own geometries, encoded in BRep format according to Open CASCADE technology [26].

The Topologic model, saved in JSON format, was represented within BTwin's KGs thanks to the '*KnolwedgeGraph.ByTBIM()*' function. This function, serving as a connector with Topologic, allowed reading the Topologic JSON format and converting it into a graph coherent with the previously defined ontology without loss of information. Specifically, the Topologic CellComplex was modeled as a BTwin 'Building', while the Topologic Cells were modeled as BTwin 'Spaces'. Furthermore, 'BuildingStoreys' and 'Zones' were created based on the information contained in the dictionaries of the Cells (during the BIM process, the 'HasLocation' relationship was encoded within the cells' dictionaries to add a reference to the building storeys and zones hosting the cells).

The elements that make up the spatial hierarchy of the case study are identified in the graph in Fig. 3 with colored nodes. In this graph, the district instance <disctrictX> contains the site instance <siteX>. A building instance, <buildingX>, is located within <siteX>. This building comprises two BuildingStoreys, <BuildingStorey_926cm> and <BuildingStorey_1418cm>, which house 66 spaces, denoted by the notations <Cell0>, <Cell1>, <Cell2>, and so forth. These spaces are grouped into eight distinct energy zones (<ZN_Classroom_A>, <ZN_Classroom_B>, etc.). For instance, the cells <Cell45>, <Cell46>, <Cell19> are included in the zone <ZN_Circulation_A>. The relationships 'HasLocation' and 'IsLocationOf' specify all connections within the spatial hierarchy.

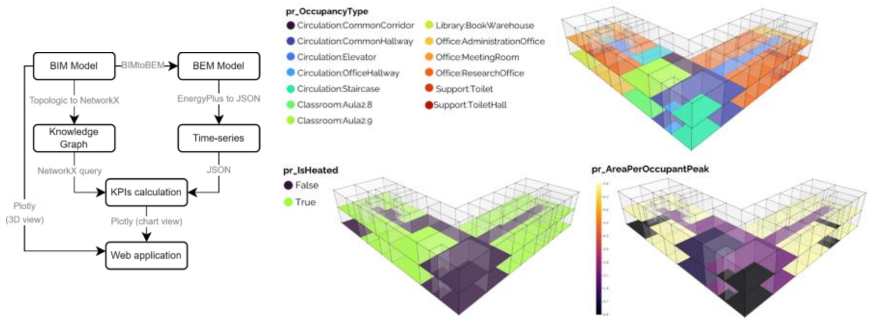

**Fig. 2.** On the left, diagram illustrating research workflow. On the right, the Topological BIM model of the selected case study building in Topologic. © 2024, Authors.

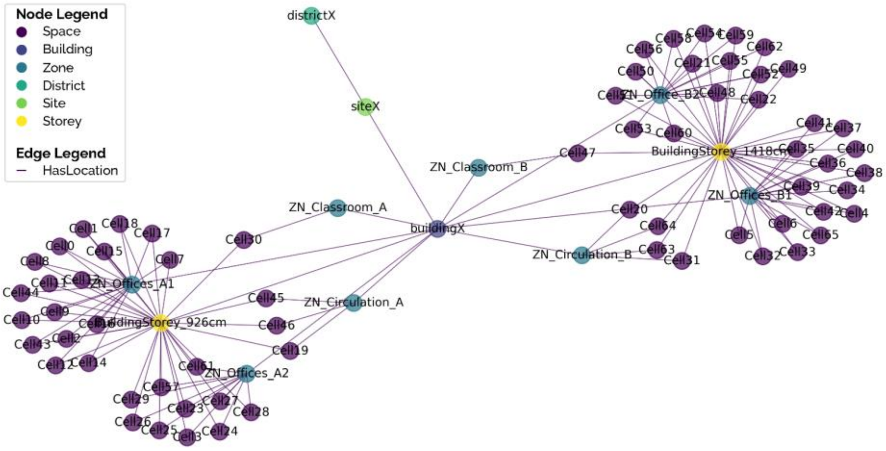

**Fig. 3.** Spatial hierarchy of the case study within a KG © 2024, Authors.

## 4.4 Property modeling

After having created the spatial hierarchy, information regarding the attributes of the various spatial elements present in the graph was added. By extracting the data already present in the Dictionaries of the Topologic model, these attributes were modeled in the BTwin graph as PSets. Specifically, the attributes set at this stage pertained to PSets related to the dimensions and occupancy of spaces, useful for normalizing the mentioned KPIs. For each space, two PSets were modeled: <ps_SpaceCommon>, which grouped dimensional properties, such as the area and volume of spaces, and <ps_OccupancyRequirements>, which grouped occupancy properties of spaces, such as maximum occupant capacity, functional typology, and occupant density. Tab. 2 summarises the properties extracted and modeled within the PSets.

**Table 2.** Properties assigned to the spaces within the KG.

| Property Name | Quantity | Unit | PSet Name | Description |
|---|---|---|---|---|
| Gross Area | Area | m² | Space Common | The total area of the space. |
| Gross Volume | Volume | m³ | Space Common | The total volume of the space. |
| Occupancy Density | Area per person | m²/pp | Space Occupancy Requirements | The area allocated per person within the space during peak times. |
| Occupancy Number Peak | People count | pp | Space Occupancy Requirements | The maximum number of people in the space during peak times. |
| Occupancy Type | Text | - | Space Occupancy Requirements | The intended function of the space (e.g., office, classroom, corridor). |

### 4.5 Simulation points

Subsequently, the focus shifted to the modeling of dynamic data.

To generate this data, the IDF model of the building was first taken, which was created from the previously described BIM model according to the procedure already detailed in [25]. This model contained all the necessary data for running simulations in EnergyPlus, including information on zone occupancy (area per occupant, number of maximum occupants, occupancy types, and occupancy schedules), dimensions of the zones (area, volume, height, and geometry), requirements for space lighting and conditioning (temperature and lighting setpoints, lighting power density, equipment power density, lighting schedules, ventilation rates, and HVAC schedules), as well as the material and thermal properties of opaque surfaces (material layers, conductivity, thermal transmittance, and volumetric heat capacity) and transparent surfaces (U-value, solar heat gain coefficient, and visible transmittance). The simulation output consisted of a CSV file containing, hour by hour, the simulation values calculated by EnergyPlus for each zone of the building (e.g., heating, cooling, lighting, and electric equipment energy demand).

Using the file as input for the 'Points.ByEnergyPlusCSV()' function, the data contained in the file was aligned with BTwin’s ontology and modeled as Points according to the described syntax, using again the connector logic. Each of these Points represented the calculated value of a certain quantity (e.g., energy needed for heating) relative to a specific hour of the year and a specific building zone. In total, 420,480 Points were modeled, corresponding to 8 Zones multiplied by 8760 hours multiplied by 6 quantities (heating energy demand, cooling energy demand, lighting energy demand, equipment energy demand, people gains, and solar gains). Each of the Points was added to the KG and connected to the respective zone.

### 4.6 Calculation of Key Performance Indicators

Zone by zone, the calculation of KPIs and the modeling of KPISets were carried out.

Four distinct KPISets– ‘Energy’, ‘Cost’, ‘Environment’, and ‘Occupancy’ – and five KPIs – natural gas demand and electricity demand for the first KPISet, energy cost for the second, equivalent CO2 emission for the third, and people heating energy for the last – were identified. Each of these KPISets was added to the graph according to the previously described semantic rules and linked to its respective zone. The addition of KPIs completed the graph database used for the demonstration presented here, resulting in the KG shown in Fig. 4.

Tab. 3 outlines the KPIs analyzed by the DDSS and their descriptions. Regarding energy needs, these were calculated by aggregating and summing the Points related to the quantities of lighting electric energy, total cooling energy, and equipment electric energy for electricity, and total heating energy for gas. Similarly, KPIs related to solar and people's internal contributions were calculated. For cost calculations, the energy data were multiplied by the unit prices of gas and electricity, derived from the building's utility bills of the last three years, which were 0.054 EUR/kWh for gas and 0.159 EUR/kWh for electricity. The conversion from standard cubic meters (smc), as indicated in the bills, to kilowatt-hours (kWh) for natural gas was achieved using a conversion factor of 10.69, taking into account the calorific value of the gas. For emissions, multiplication factors of 0.49 kgCO2eq/kWh for electricity and 0.25 kgCO2eq/kWh for gas were considered based on the greenhouse gas emissions data from the International Energy Agency (IEA) database.

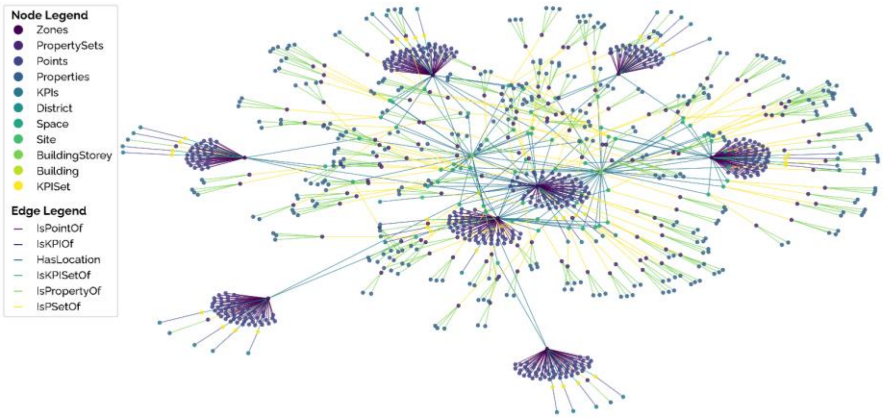

**Fig. 4.** Graph network for organizing information related to the presented case study. © 2024, Authors. To facilitate the visualization of the graph, only Points related to 10 calculation hours have been considered.

## 4.7 Dashboard development

Once all the data was stored within the graph database, specific Python functions were developed to create thematic dashboards capable of visualizing the data, both on the topological 3D model and in charts and tables. Based on user requests, these functions assign Plotly representations to a Dash application, which serves as a framework for creating shareable web-based dashboards (Fig. 5). Through the dashboard, users can select the KPIs to be displayed, choose scenarios to compare, explore the 3D model, and observe how these KPIs are distributed across various zones of the building.

**Table 3.** KPIs assigned to the zones within the KG.

| KPI Name | Quantity | Unit | KPISet Name | Description |
|---|---|---|---|---|
| Natural Gas Need | Natural gas volume | smc/m² | Energy | The volume of natural gas required for heating the space. |
| Electricity Need | Energy | kWh/m² | Energy | The amount of electrical energy needed for operating the space. |
| CO2 Emission | CO2 equivalent | kgCO2eq/m² | Environment | The emissions of CO2 equivalent resulting from energy consumption. |
| Energy Cost | Expenses | EUR/m² | Cost | The cost associated with energy consumption. |
| People Heating Energy | Energy | kWh/m² | Occupancy | The internal gains by people calculated considering the space occupancy level. |

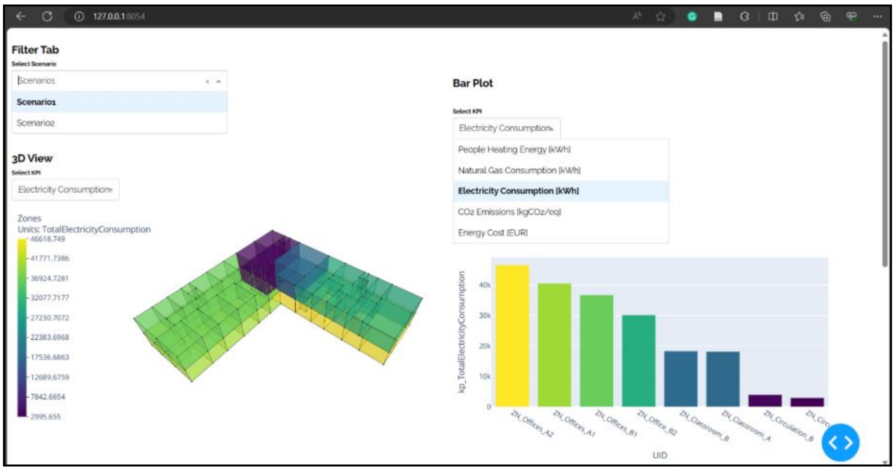

**Fig. 5.** Interactive dashboard developed for visualizing KPIs related to the case study analysis presented. © 2024, Authors.

## 5 Conclusion

Among the main goals of the digital revolution of the AECO sector, there is the aim to streamline and enhance the decision-making process for building management. Despite significant technological advancements to date, the creation of tools aimed at achieving this goal has not yet fully met current needs due to the substantial investments required.

This article introduced the BTwin software library to bridge this gap by providing a toolkit for prototyping DDSS for building management. Designed to be modular, flexible, user-friendly, and cost-effective, this toolkit leverages various Python functions to semantically link building data from different sources (e.g., BIM, BPS, meters, and sensors) within graph networks and enable facilitated data visualization through dashboard-based user interfaces. The dashboard interface is designed to be user-friendly, making it a valuable tool for both students and professionals at various levels who interact with BIM, DT, and KG technologies.

The developed tools, useful for simplified and standardized management of building information, offer two significant benefits. On the one hand, they make DDSS prototyping more accessible, enabling developers in the AECO field to code easily and quickly. On the other hand, they allow users to preview the data immediately.

The toolkit's theoretical foundation was given in the text, describing its axioms, concepts, and classes. Moreover, a pilot case study was used to demonstrate the application potential of the toolkit, which is under development and enhancement. One of the short-term goals is to enhance the toolkit with additional connections that facilitate the alignment and interoperability of the JSON data format with more complex formats, such as IFC and RDF (Resource Description Framework). This will also improve the toolkit's compatibility with BIM and web open data exchange standards. Finally, the implementation through real case studies, using live sensors and predictive algorithms, is planned to develop a full DT system.

## 6 Acknowledgements

This research was funded by the BeTwin project (‘Building Digital Twins for Built Heritage Performance-Based Management’), under the AlmaValue 2023 program of the University of Bologna and by the Ministry of Enterprises and Made in Italy, financed by the NextGenerationEU program. Moreover, the publication was realized by a researcher with a research contract co-funded by the European Union - PON Research and Innovation 2014-2020 pursuant to Art. 24, paragraph 3, lett. a), of Law no. 240 of 30 December 2010 and subsequent amendments and Ministerial Decree no. 1062 of 10 August 2021. The authors thank Prof. Wassim Jabi who supervised the realization of the BIM model presented in the paper.

## References

1. World Green Building Council (2017) Global Status Report. Accessed: Feb 16, 2024. Available: https://www.worldgbc.org/news-media/global-status-report-2017

2. European Commission (2021) New European Bauhaus. Accessed: Feb 16, 2024. Available: https://europa.eu/new-european-bauhaus/about/delivery_en
3. European Commission (2023) Built4People. Accessed: Feb 16, 2024. Available: https://build-up.ec.europa.eu/en/resources-and-tools/links/built4people-b4p
4. de Wilde P (2023) Building performance simulation in the brave new world of artificial intelligence and digital twins: A systematic review. Energy and Buildings 292:113171. https://doi.org/10.1016/j.enbuild.2023.113171
5. Lu Q (2022) Digital twins in the built environment: fundamentals, principles and applications. ICE Publishing, London
6. Boje C, Guerriero A, Kubicki S, Rezgui Y (2020) Towards a semantic Construction Digital Twin: Directions for future research. Automation in Construction 114:103179. https://doi.org/10.1016/j.autcon.2020.103179
7. Abuimara T, Hobson BW, Gunay B, et al (2021) Current state and future challenges in building management: Practitioner interviews and a literature review. Journal of Building Engineering 41:102803. https://doi.org/10.1016/j.jobe.2021.102803
8. buildingSMART International International Foundation Classes (IFC). Accessed: Aug. 29, 2021. Available: https://www.buildingsmartitalia.org/standard/standard-bs/industry-foundation-classes-ifc/
9. W3C (2021) Building Topology Ontology. Accessed: Feb 16, 2024. Available: https://w3c-lbd-cg.github.io/bot/
10. Jabi W, Aish R, Lannon S, et al (2018) Topologic - A toolkit for spatial and topological modelling. Łódź, Poland, pp 449–458
11. Balaji B, Bhattacharya A, Fierro G, et al (2018) Brick : Metadata schema for portable smart building applications. Applied Energy 226:1273–1292. https://doi.org/10.1016/j.apenergy.2018.02.091
12. Pandas - Python Data Analysis Library. Accessed: Jan. 12, 2024. Available: https://pandas.pydata.org/
13. NetworkX - NetworkX Documentation. Accessed: Jan. 12, 2024. Available: https://networkx.org/
14. topologicpy API documentation. Accessed: Jan. 12, 2024. Available: https://topologic.app/topologicpy_doc/topologic_pdoc/index.html
15. eppy - Welcome to eppy's Documentation! Accessed: Jan. 12, 2024. Available: https://eppy.readthedocs.io/en/latest/index.html
16. Dash Documentation & User-Guide. Accessed: Jan. 12, 2024. Available: https://dash.plotly.com/
17. Paulheim H (2016) Knowledge graph refinement: A survey of approaches and evaluation methods. SW 8:489–508. https://doi.org/10.3233/SW-160218
18. Chamari L, Petrova E, Pauwels P (2022) A web-based approach to BMS, BIM and IoT integration. CLIMA 2022 conference 2022: CLIMA 2022 The 14th REHVA HVAC World Congress. https://doi.org/10.34641/CLIMA.2022.228
19. Lygerakis F, Kampelis N, Kolokotsa D (2022) Knowledge Graphs' Ontologies and Applications for Energy Efficiency in Buildings: A Review. Energies 15:7520. https://doi.org/10.3390/en15207520
20. Pritoni M, Paine D, Fierro G, et al (2021) Metadata Schemas and Ontologies for Building Energy Applications: A Critical Review and Use Case Analysis. Energies 14:2024. https://doi.org/10.3390/en14072024
21. Xie X, Moretti N, Merino J, Parlikad AK (2021) Ontology-Based Spatial and System Hierarchies Federation for Fine-Grained Building Energy Analysis. Luxembourg

22. Betti G, Tartarini F, Nguyen C, Schiavon S (2022) CBE Clima Tool: a free and open-source web application for climate analysis tailored to sustainable building design. https://doi.org/10.48550/ARXIV.2212.04609
23. Ministero dell'Economia e delle Finanze (MEF) (2015) Patrimonio della PA. Rapporto tematico. Modello di stima del valore del patrimonio immobiliare pubblico.
24. Predari G, Prati D, Massafra A (2021) Modern Construction in Bologna. The Faculty of Engineering by Giuseppe Vaccaro, 1932–1935. In: Digital Modernism Heritage Lexicon. Springer Nature Switzerland AG, Cham, Svizzera, pp 233–258
25. Massafra A, Gulli R (2023) Enabling Bidirectional Interoperability between BIM and BPS through Lightweight Topological Models. Graz, Austria, pp 187–196
26. OpenCascade. Accessed: Feb. 15, 2024. Available: https://www.opencascade.com/