



Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Edile

A.a 2021/2022

Sessione di Laurea Luglio 2022

BIM-IoT Data Visualization for FM

Relatore:

Prof.ssa Valentina Villa

Candidato:

Stefania Siccardi

Ringraziamenti

Prima di procedere con la tesi, vorrei dedicare alcune righe a chi mi ha accompagnato e supportato in questi anni al Politecnico.

Vorrei ringraziare la Professoressa Villa per avermi accompagnato nella stesura di questa tesi, insieme al Professor Piantanida e all'Ingegnere Khurshid Aliev, e per avermi introdotto a questi argomenti a cui mi sono sinceramente appassionata.

Vorrei ringraziare la mia famiglia che mi ha permesso di intraprendere questo percorso formativo lontano da casa, sostenendomi sia economicamente che emotivamente in questi anni di montagne russe, tra lacrime e grandi soddisfazioni. Siete da sempre il mio punto di riferimento e spero di aver ripagato e continuare a ripagare le vostre aspettative e rendervi orgogliosi, da quando avete assecondato la mia volontà di studiare lontano da casa, in quello che consideravo il posto migliore per la mia crescita sia formativa che personale. Ad oggi posso solo che ringraziare per aver intrapreso e concluso questo percorso qui a Torino, le mie aspettative sono state più che ripagate. Ringrazio dunque mio papà, mia mamma, Rosanna, Cosimo, Giuseppe e i miei nipotini Flavio e Rebecca che ad ogni mio ritorno a casa mi accolgono come se non fosse passato un giorno dalla mia partenza. Ringrazio anche Loredana, amica di una vita che sento sempre vicina nonostante la distanza, è stata una fortuna sederci a fianco il primo giorno di liceo. Vorrei anche ringraziare tutti coloro che ho conosciuto per i corridoi del Politecnico che hanno reso questo percorso ancora più bello, in particolare Angelo e Edo, compagni di mille gruppi, studiare con voi non è mai stato noioso. Grazie anche al gruppo 1 (più uno) le storie migliori e più divertenti da raccontare di questa magistrale sono legati a voi.

Grazie ai miei coinquilini, Alberto, Angela, Rosario, Roxana che mi hanno sempre fatto sentire a casa in questi due anni, ormai vi considero come una seconda famiglia.

Infine, ringrazio te Gabriel, non potevo chiedere compagno migliore con cui affrontare questo percorso, sei la persona che più mi è stata accanto in tutti questi anni al poli, grazie per aver sempre creduto in me anche quando io non ci credevo.

Abstract

The O&M represents the most expensive phase of building's life cycle. This is due to not correct maintenance practice, which includes lack of communication between stakeholders, loss of information during buildings' life cycle and reactive maintenance practice. In particular, the last one can cause interruption of service in public buildings and rise of maintenance cost, but also the rise of energy cost since non monitored assets can consume more than expected. In addition, the poor use of BIM in operational and maintenance phase means a waste of potential information useful for FM. In this scenario it is fundamental a change of methods. In these years many new technologies have hit the construction industry, one of them made rethink the maintenance practice. It's the case of IoT technologies, a series of physical objects equipped with sensors, software and other technologies, able to connect and exchange data over a network with no need of human-human or human-computer interaction. The placing of sensors to monitor building's asset leads to the creation of a DT, a virtual digital representation of buildings in real word. The DT can contain real time data, coming from IoT technologies, and in the construction context it can be based on BIM model. Researchers have demonstrated how BIM-IoT integration is possible. This thesis' purpose is the achievement of sensors' data visualization within a BIM model for supporting FM activities. In order to do so, it will be described fundamental concept of BIM, IoT, DT and FM. After a literature review for understanding the state of art of BIM-IoT integration methods and a review of cloud-based platforms that enable BIM-IoT integration will be done. Successively, a BIM-IoT architecture will be proposed, along with the description of its components, the fault detection methodology and data transmission and visualization. These methods will be applied in the Case study section, in which sensors data will be visualized in the BIM model through the Forge application. This type of data visualization can be a support to decision-making in FM activities, since the analysis of real-time data in 3D model can detect if, when and where a failure can occur.

Abstract

La fase di gestione e manutenzione degli edifici è la più costosa dell'intero ciclo di vita dell'edificio. Questo è dovuto a cattive pratiche di manutenzione che includono mancanza di comunicazione tra le parti interessate, perdita di informazioni durante il ciclo di vita dell'edificio e manutenzione reattiva. Quest'ultima può causare interruzioni di servizio negli edifici pubblici e l'aumento dei costi di manutenzione, oltre che quelli legati al consumo energetico, dato che elementi non mantenuti possono coincidere con un aumento dei costi rispetto a quelli attesi. In più, il poco utilizzo del BIM nella fase di gestione dell'edificio si traduce in uno spreco di potenziali informazioni, utili per il FM. In questo scenario si rivela necessario un cambiamento di metodi. In questi anni molte tecnologie sono apparse nel settore delle costruzioni, una di queste ha permesso di ripensare le pratiche di manutenzione. Si tratta delle tecnologie IoT, una serie di oggetti fisici, dotati di sensori e altre tecnologie, in grado di comunicare informazioni e connettersi in una rete, senza l'ausilio di interazioni umano-umano o umano-computer. Il posizionamento di sensori per monitorare gli asset di un edificio permette la creazione di un DT, una rappresentazione virtuale di edifici presenti nel mondo reale. I DT possono contenere dati in tempo reale, provenienti da sistemi IoT e, nel settore delle costruzioni, possono essere basati sui modelli BIM. I ricercatori hanno dimostrato come l'integrazione fra BIM e IoT sia possibile. Lo scopo di questa tesi è il raggiungimento della visualizzazione dei dati provenienti dai sensori su un modello BIM, per il supporto alle attività di FM. Per fare ciò saranno descritti i concetti fondamentali di BIM, IoT, DT and FM. Dopodiché sarà condotto lo studio dello stato dell'arte di metodi di integrazione BIM-IoT e saranno esaminate piattaforme basate su cloud che ne permettano l'integrazione. Successivamente sarà proposta un'architettura per l'integrazione BIM-IoT, accompagnata dalla descrizione dei suoi componenti, della metodologia per la rilevazione di guasti e della trasmissione e visualizzazione dei dati. Questi metodi saranno applicati nella sezione dedicata al caso studio, nella quale i dati provenienti dai sensori verranno visualizzati nel modello

BIM attraverso l'applicazione Forge. Questo tipo di visualizzazione di dati può essere di supporto a processi decisionali legati al FM, dato che l'analisi dei dati in tempo reale nel modello 3D può rilevare se, dove e quando un guasto può avvenire.

Table of contents

Ringraziamenti.....	II
Abstract	III
Abstract	IV
Index of figures.....	IX
Index of tables.....	XII
Acronyms	XIII
1 Introduction.....	1
1.1 BIM	2
1.1.1 Definition	2
1.1.2 History	3
1.1.3 Applications	4
1.2 IoT	5
1.2.1 Definition	5
1.2.2 History	6
1.2.3 Architecture	7
1.2.4 Technologies	9
1.2.5 Applications	10
1.3 Facility management (FM)	11
1.3.1 Definition	11
1.3.2 History	13
1.3.3 Applications	14
1.4 Digital Twin (DT)	16

1.4.1	Definition	16
1.4.2	History	17
1.4.3	DT's architecture for FM	17
1.4.4	Technologies	19
1.4.5	Applications	20
2	BIM and IoT integration as DT for FM, literature review	22
3	Cloud-based platform for building's Data Visualization	27
3.1	Ecodomus	28
3.2	Forge Autodesk	29
3.2.1	Data Visualization API	30
3.3	Dasher 360	36
4	Methods	38
4.1	BIM-IoT architecture	38
4.2	IoT system's components	39
4.3	Fault detection methodology	42
4.4	Data transmission and visualization	44
5	Case study	46
5.1	Data acquisition	47
5.2	Data integration	47
5.2.1	Set up of Data Visualization Forge APIs	47
5.2.2	Running the Reference Application	49
5.2.3	Replacing the default model	51
5.2.4	Adding our own sensor data on the Reference Application	57

5.3 Data visualization.....	69
6 Conclusion.....	77
7 Bibliography and web references.....	79

Index of figures

Figure 1 BIM application during project life cycle	5
Figure 2 IoT Architecture	8
Figure 3 Architecture of IoT based on its components and technologies required	10
Figure 4 Hard and soft FM services.....	13
Figure 5 Maintenance strategies	15
Figure 6 Essential components to create a DT	17
Figure 7 DT architecture for FM.....	18
Figure 8 Differences between DT and BIM.....	21
Figure 9 Prototype framework	23
Figure 10 Optimisation of maintenance process stages supported by the Smart Lighting	24
Figure 11 Research methodology process ap by Kazado et al.....	26
Figure 12 Ecodomus real time data interface with IoT	29
Figure 13 Forge APIs.....	30
Figure 14 Workflow of Data Visualization API.....	31
Figure 15 Components of Data Visualization API.....	32
Figure 16 Process to view a model in the viewer.....	33
Figure 17 Hyperion Reference App Forge viewer.....	34
Figure 18 Hyperion App with sensors' data displayed.....	35
Figure 19 Graphs quantity-time of sensors' data.....	35
Figure 20 Dasher 360 placement in building life cycle	36
Figure 21 Concept of Dasher 360.....	36
Figure 22 Dasher 360 interface with sensors' data.....	37
Figure 23 BIM-IoT architecture of “IoT Open-Source Architecture for the Maintenance of Building Facilities”	38
Figure 24 Implemented BIM-IoT architecture of [30]	39
Figure 25 Kinds of faults and data collection for preventive maintenance of the FC	42
Figure 26 Integrated anomaly detection flowchart	43
Figure 27 BIM-IoT data integration and visualization approach diagram.	45

Figure 28 Case study: (a) location, (b) 3D view of the DISEG department on Revit 2022, (c) plan view of DISEG department on Revit 2022.....	46
Figure 29 Forge Portal	47
Figure 30 Sign in page on Forge portal.....	47
Figure 31 My apps page on Forge portal.....	48
Figure 32 Selection of the API enabled in the App.....	48
Figure 33 Input of app's name and description and Callback URL.....	49
Figure 34 Client ID and Client Secret of Forge App.....	49
Figure 35 Forge app with the default model at http://localhost:9000 static address.....	51
Figure 36 Reference application directory structure.....	51
Figure 37 Creation of rooms in Revit 2022	52
Figure 38 Forge extension for VS Code.....	52
Figure 39 Insertion of FORGE_CLIENT_ID, FORGE_CLIENT_SECRET and region on setting.json in the VS Code Forge extension	53
Figure 40 Viewer in the IDE through Forge extension for VS Code with the DISEG model	53
Figure 41 Set up of environmental variables from advanced setting on windows.....	54
Figure 42 URN object.....	55
Figure 43 Setting of environmental variables on .env file	55
Figure 44 http://localhost:9000/upload	56
Figure 45 DISEG's model displayed in the Forge Viewer on the static address http://localhost:9000	56
Figure 46 Modified code to associate heatmaps to rooms.....	57
Figure 47 Rooms elements in Forge extension of VS Code	57
Figure 48 Code of device model properties.....	63
Figure 49 localhost:9000/playground with our model.....	63
Figure 50 localhost:9000/playground with sensors' coordinates in JSON format	64
Figure 51 Devices.json with coordinates of a sprite.....	64
Figure 52 Placed sprites	65

Figure 53 Line 30 of the code at “forge-dataviz-iot-reference-app\server\router\DataAPI.js”	
.....	65
Figure 54 Environmental variables for the CSV file.....	66
Figure 55 Code snippet for defining sensors' Propertyiconmap in Dot.jsx.....	67
Figure 56 Code snippet in sensorstyle.js for defining sensors' image of Propertyiconmap	
.....	68
Figure 57 Icon displayed on Forge in the “sensor list”.....	68
Figure 58 Code snippet associating icon to each parameter in “SensorStyles.js”	68
Figure 59 Code snippet for defining colours of parameters' gradient	69
Figure 60 Parameters' gradient for heatmap.....	69
Figure 61 Sensor list with sprites divided by level and room in which are placed	70
Figure 62 Graphs' parameter-time.....	75
Figure 63 Preview of data	75
Figure 64 Value at a specific time	76
Figure 65 Heatmap variation for temperature on Autodesk Forge.....	76

Index of tables

Table 1 Sensors' specification and allocation in the building.....	40
Table 2 Sensors employed and embed systems.....	40
Table 3 Sampling frequency and FC sensor allocation Source: [30].....	44
Table 4 FC features Source: [30]	46

Acronyms

AEC - Architecture, engineering and construction

API - Application Program Interface

AR - Augmented Reality

BDS - Building Description System

BIM - Building Information Modelling/model

BIM-IoT - Building Information Modelling to Internet of Things

BMS - Building Management Systems

CMMS - Computerized Maintenance Management Systems

COBie - Construction Operations Building Information Exchange

BPM - Building Project Model

CAD - Computer Aided Design

CAM - Computer Aided Manufacturing

CDE - Common Data Environment

CPS - Cyber Physical System

DT - Digital Twin

FC - Fan coil

FM - Facility Management

GLIDE - Graphical Language for Interactive Design

IoT - Internet of Things

JSON - Java Script Object Notation

Li-Fi - Light Fidelity

MQTT - Message Queue Telemetry Transport

NFC - Near Field Communication

NPM - Node Package Manager

O&M - Operation and maintenance

RFID - Radio Frequency Technology

Rpi3B - Raspberry Pi 3B

SaaS - Software as a Service

UI - User Interface

URN - Uniform Resource Name

WSN - Wireless sensor network

1 Introduction

Nowadays the Operational and maintenance phase represents the most expensive phase of building's life cycle. This is due to outdated maintenance practice, often based on paper documentation, which doesn't allow interoperability and cause lack of communication between stakeholders. Shared information on accessible platform and the implementation of BIM in the O&M phase are necessary to overcome these limits. Many technologies have arisen in the past decades connected to the collection and sharing of data and are called IoT technologies. These are, in turn, at the base of Digital twin.

The creation of building's digital twin represents an opportunity to improve and optimize the FM practice, allowing predictive maintenance, based on monitoring assets' data and detect anomalous behavior way before failures happen.

In this context the study of methods that enable the integration of sensors' data in BIM model is needed. Many studies have been carried on the topic and proposed a solution for optimizing FM practice with the help of BIM, IoT and DT.

The aim of this thesis is the achievement of sensors' data visualization within a BIM model to support FM activities. A BIM-IoT architecture for data acquisition and visualization is proposed and then applied in the case study section.

The section 1 is focused on defining the concepts, which are object of study of this thesis. The section 2 contains a literature review of studies regarding the BIM-IoT integration. Many researchers have studied methodology to make this integration possible and the more relevant will be exposed.

The section 3 investigates some cloud-based platform and describes the one which will be used for this thesis, Autodesk Forge.

The section 4 describes the methods that will be applied in the case study. In particular, it's proposed a workflow for BIM-IoT integration based on the use of BIM model and the Forge app, combining 3D model with real-time or historical data.

The section 5 represents the application of methods in the case study, which is the DISEG department at Politecnico di Torino. In this section it is shown how to customize the Forge application, especially, how to upload your own model and visualize the sensors' data.

Eventually the results of the application are discussed.

1.1 BIM

1.1.1 Definition

The ISO 29481-1:2010(E) defines BIM as a: “shared digital representation of physical and functional characteristics of any built object [...] which forms a reliable basis for decisions”.

The PAS1192-5:2015 gives the following definition: “discrete set of electronic object-oriented information used for design, construction and operation of a built asset”. The software house Autodesk considers BIM as a “holistic process of creating and managing information for a built asset”.

Several definitions have been given to BIM, but it is based on three fundamental concepts which are the words the acronym stands for: Building, information and modelling /model /management. Building does not refer specifically to the building construction, since it can be used also for infrastructure such as railways, tunnels, bridges and other utilities, but it can be seen as a reference to the verb “to build”. Information represents the core of BIM, seeing as it contains geometrical and non-geometrical information, documentation and drawings [1]. The last word of the acronym can change since it can refer to different functions of BIM:

- Modelling stands for the process that leads to the creation of the 3D model
- Model stands for the digital description of all aspects of the construction work [2].

The model is a resource of information that are shared making it an affordable source.

- Management indicates the organization and the control of the business process using the information of the model to support the exchange of information during the project life cycle.

1.1.2 History

BIM is the result of an evolution that began in 1957. In this year Dr. Patrick J. Hanratty developed the first commercial CAM (Computer Aided Machine). It was the born of CAD since CAM and CAD branches converged in one. In 1961 Hanratty developed DAC (Design Automated by Computer), which was the first CAM/CAD system that used interactive graphics [3]. The first graphical software dates to 1963, it was designed by Ivan Sutherland and was called “Sketchpad”. He created a program in which the user could graphically interact using a screen. A light pen to draft and a set of buttons which permitted to enter parameters. Even if it didn’t arrive in the commercial market, it became highly influential for future CAD developments [4].

The concept of BIM was created by Professor Eastman in early 1970s, who proposed the Building Description System (BDS) which can be considered as the precursor of BIM. The system, according to [5], was designed to perform few functions very similar to the ones that are now enabled in BIM, such as:

- Graphical user interface to store and display complex element shapes;
- An interactive graphic language to arrange e different elements in a design space;
- it could produce perspective or orthographic drawings;
- the database could be sorted by attributes such as material type.

In 1977 the Professor Eastman created Graphical Language for Interactive Design (GLIDE) and had all the characteristics of modern BIM platform [3]. In 1989 Building Project Model (BPM) was introduced with more features like estimation [5]. In 1995 the Generic Building Model was introduced, which enabled to rise the degree of construction management with information usable in the project life cycle. In the early 2000s among the architecture, engineering and construction (AEC) industry the

concept of BIM took off, when 3D parametric modelling was developed [5]. In 2009 McGrath-Hill Construction report identified BIM as a potentially transformational approach to design and construction” [6]. In 2016 the use of BIM was made mandatory for all public sector projects from 2016 in UK [6]. This rose even more the attention from the AEC industry to the BIM standards and protocols.

1.1.3 Applications

BIM’s applications in AEC industry are wide and can cover all project’s life cycle. It can be considered four basics steps that all construction projects face and are: plan, design, construction and operation and maintenance (O&M). BIM can be applied in all this phases.

In the plan phase BIM permits to program the execution of the project based on site condition analysis result, phase planning and estimated budget of the entire project. It is also useful for the communication of design ideas [7].

In the design phase BIM permits the quick visualization of perspective, sections, plan views and quantity take off can be quickly and automatically be visualized and updated depending on changes made on the model. Besides engineering analysis, such us energy and structural, can be performed. Since the model includes architectural, structural and also pipeline’ systems, for example, clash detection can be performed, which permits to identify if, where or how two parts of the building interfere with one another.

In the construction phase thanks to BIM, simulation of construction process can be performed. Besides objects in the model can be grouped according to the phase in which they are built or demolished and linked to activities in projects scheduled.

In the O&M phase the information stored in the BIM model can be useful for later access and retrieval and can help FM activities.

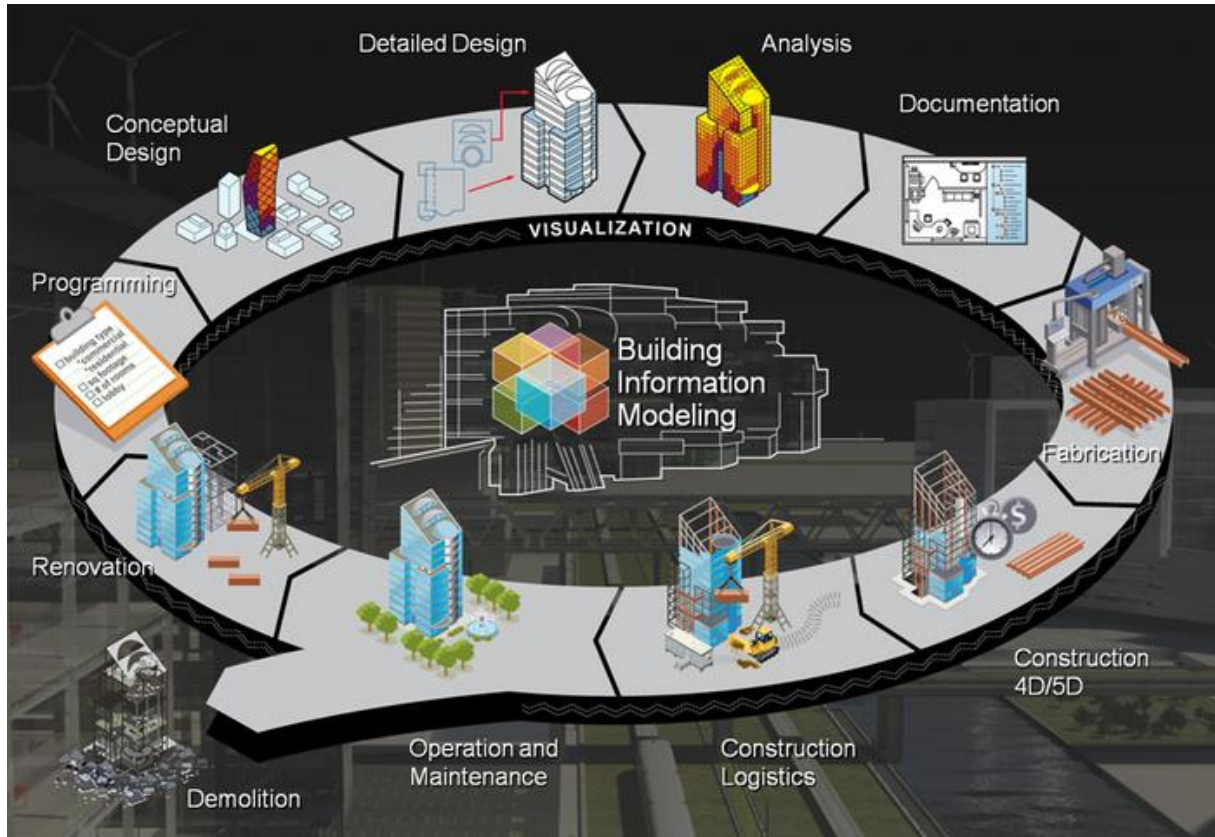


Figure 1 BIM application during project life cycle

Source: [8]

1.2 IoT

1.2.1 Definition

“An open and comprehensive network of intelligent objects that have the capacity to auto-organize, share information, data and resources, reacting and acting in face of situations and changes in the environment” [9].

The term Internet of Things (IoT) identifies a series of physical objects, which are equipped with sensors, software and other technologies, able to connect and exchange data over a network with no need of human-human or human-computer interaction [10].

This term was coined back in 1999 by Kevin Ashton, the executive director of the Auto-ID Center, who used it as a title of a presentation at Procter & Gamble [11] [12]. What he meant with this expression was that almost all the computer devices today are dependent on information given by humans, and that’s an issue for many reasons. In the first place, humans are not very efficient and accurate at capturing data from things in

the real world. Instead, if computers had the capacity of capturing information and data on their own, we would be able to track and count everything with a reduce of cost, loss and waste [11].

IoT is composed by two words: internet, which indicates a global system of interconnected computer networks that use the standard internet protocol suit and things that are objects or persons present in the real world [9].

The fundamental concept that stands at the center of the IoT technology is the connection between objects and the exchange of a wide amount of information and data, without any intermediary, since the data comes from sensors embed in the object itself.

1.2.2 History

Even if the term was coined in 1999, the first implementation of this technology dates back to early 1980s in a Coke machine at Carnegie Melon University [9]. Programmers were working at a floor above the one where a Coke machine was placed, so they wrote a server program that checked how long a column of the vending machine was unfilled, in order to decide whether to go at the floor of the vending machine for a cold drink or not.

In 1999 Neil Gershenfeld wrote a book called "When Things Start to Think" on the topic and Neil Gross in a Business Week's article affirmed:

"In the next century, planet earth will don an electronic skin. It will use the Internet as a scaffold to support and transmit its sensations. This skin is already being stitched together. It consists of millions of embedded electronic measuring devices: thermostats, pressure gauges, pollution detectors, cameras, microphones, glucose sensors, EKGs, electroencephalographs. These will probe and monitor cities and endangered species, the atmosphere, our ships, highways and fleets of trucks, our conversations, our bodies--even our dreams." [13]

In 2000 the company LG announced its first Internet refrigerator plans [13]. In 2002 David Rose and others created Ambient Orb, a device able to monitor the Dow Jones,

personal portfolios, weather and other data and changes color according to the dynamic parameters [13].

In 2003-2004 the term appears on The Guardian, Scientific American and the Boston Globe [13]. In these years many projects on the topic arise and give implementations to the first ideas of IoT. Meanwhile The Us Department of Defense employ RFID on a massive scale [13]. In 2005 the UN's International Telecommunications Union ITU published its first report on the topic [13]. In 2006-2008 there is the recognition by the European Union and the first European IoT conference [13]. The born of IoT for Cisco Internet Solutions Group (IBSG) is set between 2008 and 2009, when more "thing or objects" were connected to the internet than people [13]. Since then, some countries like US and China started investments in this technology. Also major tech companies like IBM, Cisco, Ericsson contributed to the growth and the use of IoT [13].

1.2.3 Architecture

The IoT architecture has changed in the past years in order to satisfy IoT development. At the beginning there was a 3-layer architecture that consisted of these three layers:

- **Perception layer:** it identifies each object of the IoT system and includes RFID tags, sensors, cameras;
- **Network layer:** it is the main point of the IoT, since it permits the transmission of information coming from the perception layer, it includes the software and a hardware instrumentations of internet network and also the management and the information centers;
- **Application layer:** it represents the fields of application of the IoT technology in order to fulfill human needs. [14]

This three-layer architecture has been overcome by the five-layer architecture that is composed by:

- **Perception layer:** also called "Device layer", because it includes the physical objects and the sensor devices. The sensors, according to the identification method of the object, can be RFID, 2D-barcode or infrared. The task of this layer is to identify the objects and collect the data coming from them;
- **Network layer:** it is the bridge between the collected data and the information processing system. It transfers data from the sensors to the system through wired or not wired technology such as Wi-fi, Bluetooth, Infrared, 3G etc. Its principal purpose is to transfer information from the perception layer to the middleware layer;
- **Middleware layer:** it executes information processing ubiquitous computation and takes automatic decision based on the results [15];
- **Application layer:** this layer is responsible for the management of the application based on the objects' information processed in the Middleware layer [15];
- **Business layer:** it handles the management of the IoT system. This layer permits the elaboration of the data collected through the creation of graphs, business models and flowcharts etc. The business layer represents the base to decide future actions and business models.

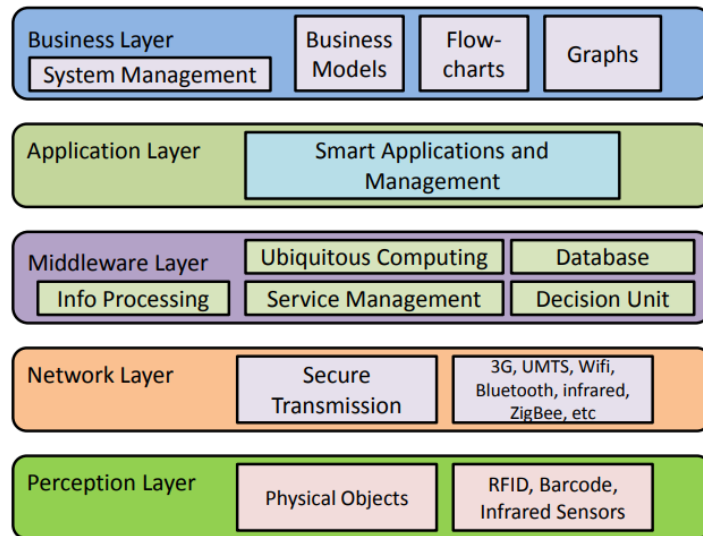


Figure 2 IoT Architecture

Source: [15]

1.2.4 Technologies

An IoT devices needs to have some specific features [12] [16]:

- Be uniquely identified;
- Be able to collect and exchange data;
- actuate devices based on triggers;
- assist in communication.

In order to fulfill these requirements, the key IoT technologies are Radio Frequency Identification (RFID), the sensors technology, nanotechnology, intelligence embed technology [17].

The RFID technology is used to identify in a unique way the object. It is a system that transmits the identity of an object or person wirelessly using radio waves in the form of a serial number [18]. The main components of RFID are tag, reader, antenna, access controller, software and server [9]. But the RFID technology is not the only one that can be employed to give a unique identification to the objects. In fact 2D-barcode, Wireless Sensor Networks (WSN), Quick Response Code (QR code) etc. are also employed for the purpose.

The communication technologies used are:

- Zigbee: commonly used in home automation, it has a small range protocol of about 20 meters and it's used to create a small network [19];
- Z wave: it's a long range wireless protocol (around 100 meters) [19];
- MQTT (Message Queue Telemetry Transport): is a machine-to-machine IoT protocol and its main feature is the ability to transmit information from one source to many users via an intermediate node [19];
- Bluetooth: it's a short range protocol;
- Li-Fi (Light Fidelity) is a short-range wireless protocol, where the transfer of data takes place in the form of light [19];
- Wi-fi: it's a medium range network employed in local area network [19];

- Near Field Communication (NFC): is a really short-range networking protocol (4 meters) and provides point to point connectivity between communicating devices [19];
- HaLOW: similar to Wi-Fi, but with a lower data transfer rate and is a medium range protocol [19];
- Power Line Area Network: is a long range wired communication network and uses power lines for transmitting data [19].

These all technologies of IoT makes use of either IPv4 or IPv6 for addressing [19].

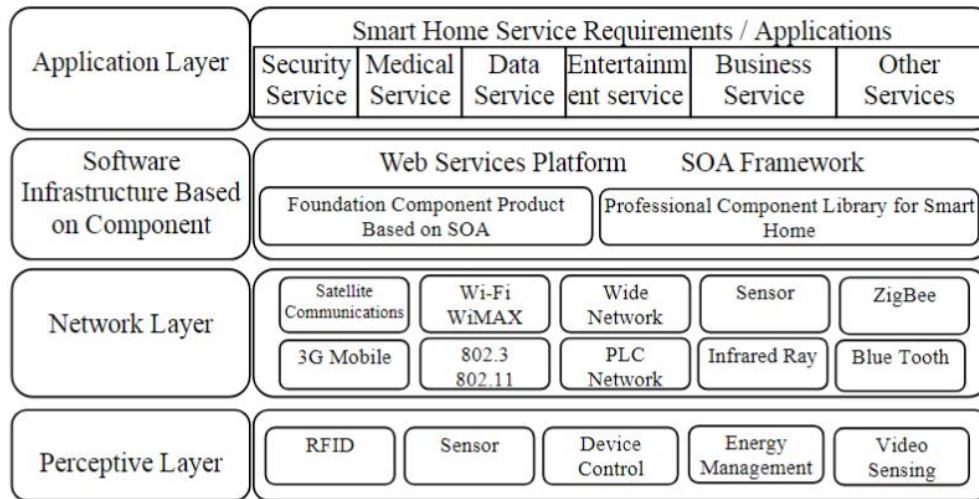


Figure 3 Architecture of IoT based on its components and technologies required

Source: [17]

1.2.5 Applications

The application fields of the IoT technology are wide but only a small part is currently available to our society [20]. This technology can be used in the health care system to track and collect data about patients in order to monitor health parameters and avoid errors about drug/dose/time/procedure of drugs administrations for example [20]. Also traffic could take advantage introducing a traffic monitoring system that enables identification of vehicles and other traffic factors, in order to avoid congestions, make available

data of parking spaces, reduce environmental pollution and have vehicle violation monitoring and thief detection [16] [15].

Monitoring of environmental data could help the prevention of environmental disasters [15] [16].

In the construction industry the IoT technology is used with the integration of BIM. In fact this integration” offers complementary views of the project that together supplement the limitations of each” [21]. BIM models represent the building in an accurate way geometrically and with the IoT technology it could also display information about position, physical measurements, weather etc. [21]. BIM and IoT devices can be employed in energy management, construction monitoring, health and safety management and building management, but the research is still at an embryonic state and most studies are theoretically and conceptually proposed [21] [22].

1.3 Facility management (FM)

1.3.1 Definition

According to IFMA FM is “a profession that encompasses multiple disciplines to ensure functionality, comfort, safety and efficiency of the built environment by integrating people, place, process and technology”.

According to Yalcinkaya et al. FM is “a discipline comprising of various operations, activities and maintenance services to support the main functions of an in-use building or facility” [23].

All these definitions don’t seem to agree on what really is FM, since they defined it as a “discipline”, or a “profession”, but all of them point out at the same purpose which is the conservation and improvement of the building’s efficiency and the conservation of its

main functions. In fact, the construction industry FM mainly deals with maintenance of structures.

So it can be affirmed that FM is concerned with the management of facilities in the built environment at both a strategic and a day-to-day level to deliver operational objectives and to maintain a safe and efficient environment [24].

The services that FM includes are wide and involve:

- Maintenance and operations such as cleaning and security;
- Energy management;
- Workspace and occupant management;
- Lease management, administration and accounting;
- Project planning and management;
- Staff and user experience;
- Compliance;
- Emergency management and business continuity;
- Integrated FM and Property/real estate management.

FM can be divided into two basic areas, which are hard and soft FM services. The first concern the maintenance and management of any physical part of a building, including assets, space, and infrastructure and includes [25]:

- Management of planning, construction, design, and relocation projects
- Management of building systems including HVAC, electrical, and plumbing
- Real estate management and leasing
- Preventive maintenance (PM) on buildings, interiors, and assets
- Managing and responding to maintenance requests
- Other capital improvements

The soft FM services are related to people make facilities more comfortable, satisfying, and secure [25]:

- Building security

- Space planning
- Responding to environmental, health, and safety issues, including emergency planning and preparedness
- Catering and food services
- Cleaning, sanitation, and janitorial services
- Groundskeeping, landscaping, and pest control
- Educating others about regulations and compliancy requirements
- Mail management
- Waste management



Figure 4 Hard and soft FM services

Source: <https://ftmaintenance.com/maintenance-management/what-is-facility-management/>

1.3.2 History

The FM has 30-40 years of existence. The roots of FM in USA dates back to 1978 when Herman Miller Research Corporation hosted a conference on “Facility Influence on Productivity” and this brought to the creation of the International Facilities Management Association (IFMA) an organization specialized in FM, which still exists and now represents a reference point for professionals in the FM. In Europe FM emerged 25 years

ago when there was a need of “linking real estate and construction industry concerns with the productive use of workplace building assets in light of the fact that in those days, at least 25 percent of the assets of the corporate balance sheets of their companies were tied up in real estate assets requiring effective management” [26].

3.2.3 Applications

FM has become an important activity to decrease the cost in business operation and increase efficiency. Studies demonstrates that the O&M phase is the longest and more expensive phase of building life cycle and significantly affects the total life cycle cost of the building with an incidence of the 60% [27]. It is important to highlight that a major part of the O&M phase’s costs come from the inadequate or even absent interaction between stakeholders and roles involved in the building’s life-cycle [27]. Most of the buildings’ information is based on paper documents, which implicates a waste of time in searching useful information and is responsible of the decrease of efficiency in the O&M phase.

In this context stands the relevance of a correct practice for FM, with a need of a different approach from the traditional one.

The first important requirement that arise from the flaws of the traditional paper-based approach is the sharing of accurate and reliable information, accessible at any time.

Many studies have found the solution in the implementation of BIM in the O&M phase. Despite its great usage in the design and construction phase, thanks to its visualization and coordination capabilities, it’s still not very much used in the O&M phase. Researchers are testing its great potential with other technologies to support FM activities. Indeed BIM is a great container of information but needs implementation for FM usage, since it’s a container of static information, but inadequate, for example, to monitor building’s facilities during their life cycle. The need of updated and real time data for FM activities has brought researchers to find a solution in the integration of BIM with IoT technologies. The use of IoT technologies with BIM permits the gather of real time data, but also the

creation of a reliable historical database. All these practices have brought to the advent of a new type of a maintenance: predictive maintenance. Maintenance approach can be divided into three main groups:

- Corrective maintenance, also known as reactive maintenance or run to failure maintenance it consists in intervening after the failure;
- Preventive maintenance, which consists in carrying out maintenance and inspections actions to prevent the occur of breakdowns. It can be time based or usage based. Even if it prevents breakdowns it is based on inspections that can be useless, but represent a cost and also failure can still occur;
- Predictive maintenance consists in monitoring data to predict the future machine health state. The purpose of this approach is to predict when, where and which components may have potential failures.

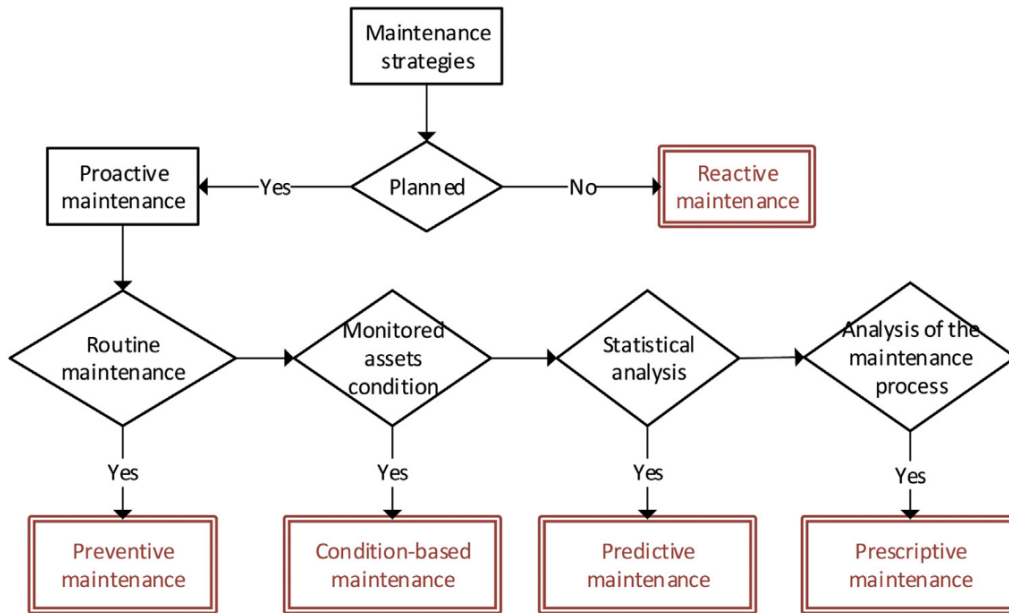


Figure 5 Maintenance strategies

Source: [28]

Nowadays most of maintenance's practice consist of corrective and preventive maintenance, whereas predictive maintenance is used only in particular cases [29]. Predictive maintenance is an opportunity for the FM sector to reduce unplanned failures, reduce

maintenance cost and penalties, through the use of sensors and machine learning algorithms [30] [29]. The IoT technology makes possible to collect real time data through sensors and visualize them in BIM based dashboard.

1.4 Digital Twin (DT)

1.4.1 Definition

The first definition was given by Grieves in 2003 during his lecture and it was described as a: “virtual digital representation equivalent to physical products” [31]. According to [32]: “in 2012 NASA gives a definition of DT in their integrated technology roadmap (Technology Area 11: Modeling, Simulation, Information Technology & Processing Roadmap; 2010) which has been adapted in: “A DT is an integrated Multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin” “. In 2015 Rios et al. [33] substituted the word “vehicle” with “product” for a more general definition [31]. Various definition were attributed to DT during years on the basis of the industrial fields where it was used [31]. In general DT can be defined as “the digital replica of physical assets, processes, people, places and systems, which provides both the elements and the dynamics of how the complex system operates and evolves throughout its lifecycle.” [31].

In the construction context the DT of a building, according to researchers, can be defined as: “the interaction between the real-world indoor environment of the building and a digital but realistic virtual representation model of the building environment, which provides the opportunity for real-time monitoring and data acquisition.” [34]. In this definition real world indoor environment stands for information about temperature, relative humidity, airflow and lighting conditions whereas digital virtual environment stands for computational fluid dynamics and luminance levels [30].

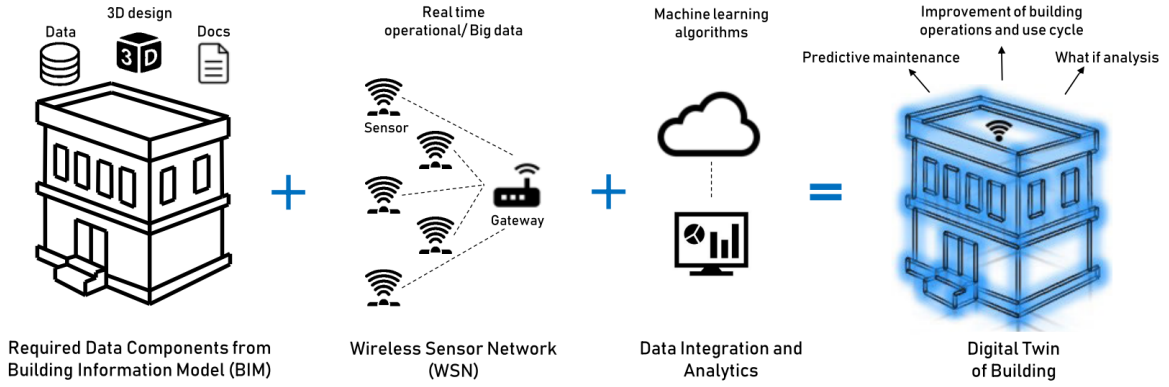


Figure 6 Essential components to create a DT

Source: [35]

1.4.2 History

The first application of DT dates back to 1970s when National Aeronautics and Space Administration (NASA), during the Apollo Program, built a replica of space vehicles on Earth to mirror the condition of the equipment during the mission [31].

According to [31] in 2003 Michael Grieves presented a lecture on product-life cycle management where the use of DT was proposed. In 2012 NASA applied the DT "to integrate ultra-high-fidelity simulation with a vehicle's on-board integrated vehicle health management system, maintenance history, and all available historical and fleet data to mirror the life of its flying twin and enable unprecedented levels of safety and reliability" [31]. The born of IoT technology helped the development of DT in the manufacturing industry, in fact enterprises started to develop platform of DT for real-time monitoring, inspection and maintenance [31].

1.4.3 DT's architecture for FM

The DT in order to be useful in the FM phase needs to have specific features which includes efficiency, interoperability, intelligence and integration [36].

Xie et al. [36] proposed an architecture for DT to accomplish FM requirements, which is composed by five layers:

- Data acquisition layer,

- transmission layer,
- digital modelling layer,
- data/model integration layer
- service layer

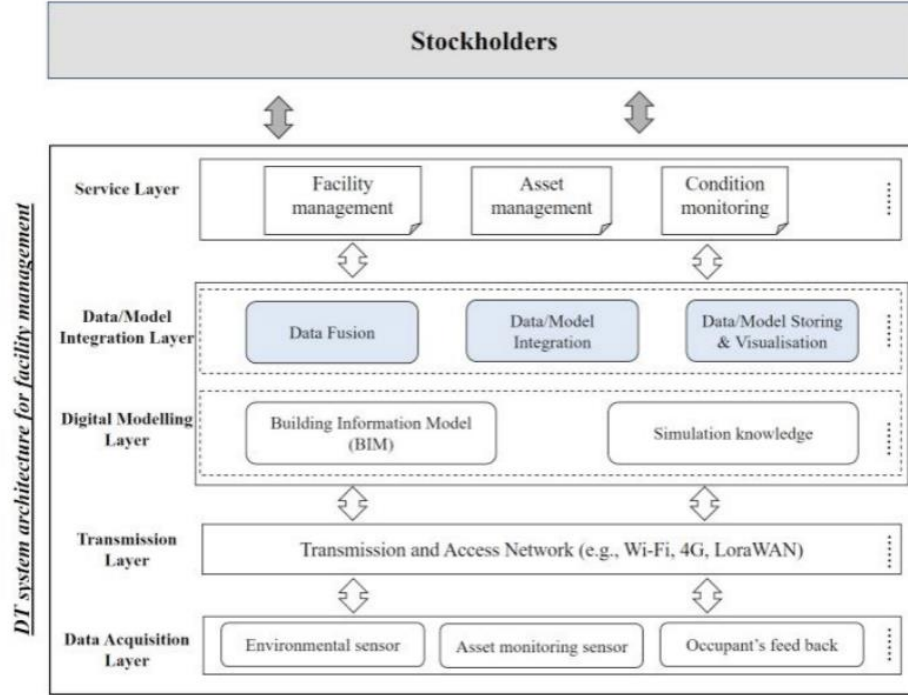


Figure 7 DT architecture for FM

Source: [36]

The data acquisition layer is the most important for every DT.

The data gathering represents a cost so only the essential data have to be collected. In this case the IoT technology are an important tool to collect data because of their cost-efficiency, low power consumption and elimination in maintenance need [36].

The transmission layer's purpose is the transfer of data coming from higher layer for modelling and analysis. To do so "variety of communication protocols could be adopted according to the transmission range, data speed and data volume required in the specific scenario" [36].

Digital modelling layer concern BIM usage, since it represents a database for geometrical and non-geometrical data of assets' life cycle. This information are important for O&M phase and this knowledge can be much easier to manage, share and exchange. In this layer are included also simulation engines which gives environmental and energy analysis, structure analysis and others.

The data model integration layer integrates all the data, simulations and domain knowledge through BIM and analyses and updates the as-is condition of the assets, processes and systems [36].

The top layer is constituted by the service layer, which can be seen as a translator of information for the end user and enables the interaction between stockholders and DT system. This layer can be seen as a separator between users and technical details, which facilitates the usability of the DT system [36].

1.4.4 Technologies

The concept of DT needs to be supported by some technologies to make it possible. Liu et al. [37] described these technologies from three perspective:

- Data related technologies
- high fidelity modeling technologies
- model based simulation technologies.

Data are at the base of the concept of DT. This are collected thanks to the employ of sensors, RFIDS tags and readers, gauges, cameras, scanners. This great amount of data collected need to be transmitted at high velocity. This implies the use of 5G to transmit real time data and edge computing can be useful to pre-process the data gathered and reduce the network burden and also avoid data leakage. "Data mapping and data fusion are also needed to understand the collected data. The most common data mapping technology used in literatures is XML" [37].

High fidelity modeling technologies are based on Multiphysics modelling, which is essential to integrate semantic data models and physical models. According to [37]: “Semantic data models are trained by known inputs and outputs, using artificial intelligence methods. Physical models require comprehensive understanding of the physical properties and their mutual interaction”.

1.4.5 Applications

Application fields of DT are various, since it improves the product, the design and the production and reduces logistic risks [38]. Also, it predicts process behaviors [38]. In the manufacturing industry is used for predictive maintenance and to optimize and improve the production speed. In the aviation field is mainly used for predictive maintenance, decision support, optimization and diagnostic. In the healthcare environment it was firstly used for predictive maintenance of the medical devices and their performance optimization, but later it was involved in the optimization of the hospital life cycle, like management and coordination of patient care and afterward the objective is to create the DT of humans in order to prevent illness thanks to the monitor of real time parameters and historical data [39].

In the building context, according to [38], the DT definition could seem similar to the BIM definition, but they differentiate in vary aspects like purpose, technology, the end-users, and the facility life stage. One of the main differences between BIM and DT is that the first can contain static information, while DT is a dynamic model, able to monitor the physical assets of the building with real time data. This collocates the use of BIM and DT in different phases of the building life cycle. In fact BIM is used mostly for the design phase, while DT can be used in the O&M phase, since it enables facility managers to actuate predictive maintenance and helps in well informed decision-making [35] [38]. The data collected during the O&M phase by the DT can be also used by engineers and architects to improve the design of future building based on the detected flaws [35].

Differentiator Concept	Application focus	Users	Supporting technology	Software	Stage of life cycle	Concept origin
BIM	Design visualization and consistency, Clash detection, Lean construction, Time and cost estimation, Stakeholders' interoperability [10]	AEC, Facility manager [16], [30]	Detailed 3D model, Common data environment (CDE), Industry Foundation Class (IFC), Construction Operations Building Information Exchange (COBie) [16]	Revit, MicroStation, ArchiCAD, Open source BIMserver, Grevit [16]	Design, Construction, Use (maintenance), Demolition [31]	Charles Eastman [16]
Digital Twin of Building	Predictive maintenance [26], Tenant comfort enhancement, Resource consumption efficiency, What-if analysis, Closed-loop design [23]	Architect, Facility manager	3D model, WSN, Data analytics, Machine learning [32]	Predix, Dasher 360, Ecodomus	Use (operation) [33]	NASA's Apollo program [22]

Figure 8 Differences between DT and BIM

Source: [35]

The contribution of DT in the O&M phase can be divided in three categories according to [40]:

- Monitoring: the data are collected and used to update the virtual parts from the physical parts, including defect detection and asset monitoring;
- Analysis: the data collected in the virtual parts are examined for analysis, diagnose and decision making;
- Action: it includes doing something with the physical parts using virtual parts, such as automatic control, retrofitting and demolishing.

Focusing on geometric and non-geometric information, the DT employs sensors to upgrade the data in time for the accounting of virtual parts from physical parts to realize asset management.

2 BIM and IoT integration as DT for FM, literature review

The BIM, IoT and DT technologies can be all considered part of the Construction 4.0, whose first mention dates back to 2016 when construction companies understood the importance of digitalization in the construction industry [41]. The Construction 4.0 can be described as a paradigm that includes Cyber Physical System (CPS) and the IoT data and services, with the purpose of creating a digital construction site and connecting the digital layer, such as BIM and Common Data environment (CDE), with the physical layer constituted by the asset and its life cycle [41].

In this section the state of art BIM, IoT and DT for FM will be described.

BIM with IoT technologies permits to create a DT of a building, with real time data that change the approach to maintenance from reactive maintenance towards predictive maintenance.

According to [30] the benefits for FM, originated by these technologies, includes: “document management, historical data cataloguing, logistics and material tracking, building component life cycle monitoring, and building energy controls”. Besides it has been studied how DT and IoT can support decision making for the final user, permits a better data collection and communication, predictive maintenance and scheduling [30]. All these improvements allow to achieve a save of cost and time in the O&M phase. In this context BIM represents a repository for project’s information and a bridge between all the stakeholders involved, but at the actual state it is not used at its full potential in the O&M phase. Even if BIM is a storage of data, often it is not updated during the life cycle and does not represent the building as built, besides the data could not be useful for FM since the facility managers do not participate in the creation of the model [30]. Moreover, the lack of open systems and interoperability between BIM and FM technologies represent obstacles.

Many studies tried to integrate these technologies to overcome these limits.

Fialho et al. [42] studied a prototype that included BIM and IoT-based smart lighting maintenance system to support university FM sectors on decision-making. The purpose of the prototype proposed was the accurate and anticipated transfer of the lighting performance to support university maintenance teams in making decisions. The prototype's scope included:

- Automatically detect and report current and developing failure by generating notification messages triggering maintenance service response;
- Automatically populate real-time information from the sensor's lighting central database into the SmartLab app and the BIM model parameters;
- Visualise updated lighting information on BIM and IoT interfaces.

The structure of the prototype by Fiahlo et al. (2022) is displayed in the following figure.

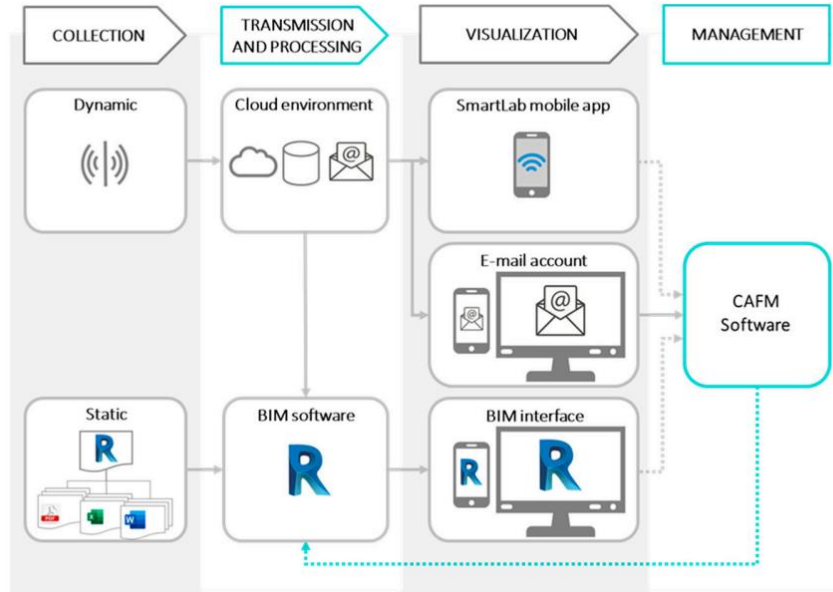


Figure 9 Prototype framework

Source: [42]

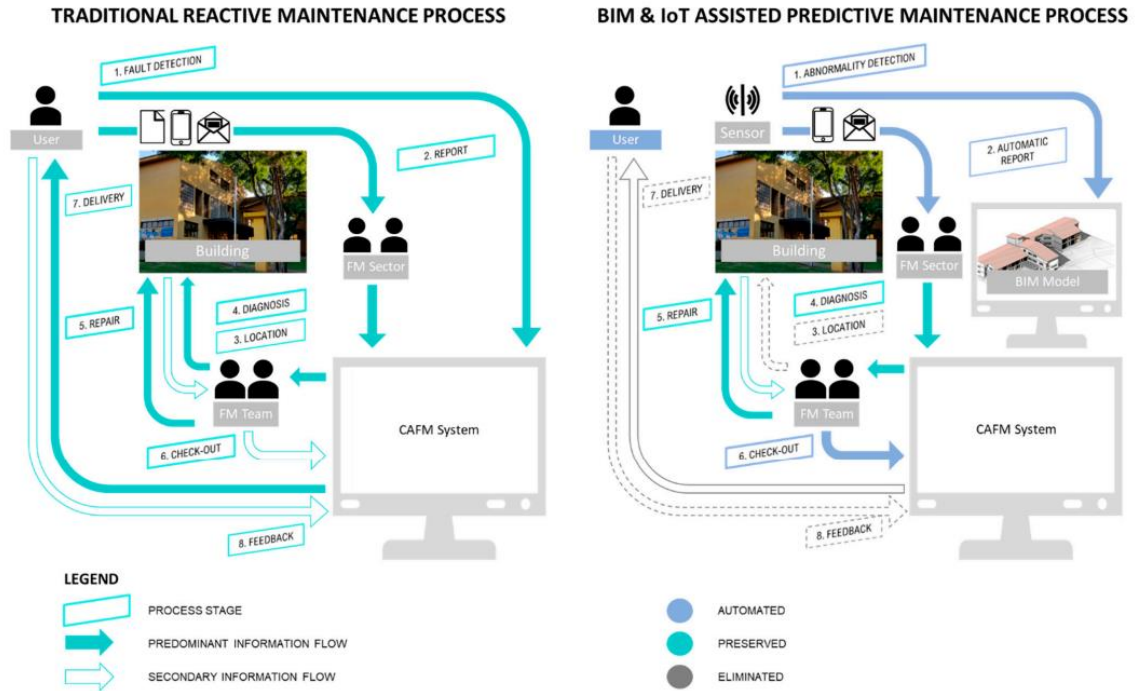


Figure 10 Optimisation of maintenance process stages supported by the Smart Lighting

System prototype

Source: [42]

This study proved the necessity of an identification for technical, functional and FM BIM information requirements for an effective smart system implementation, so that the communication between stakeholders becomes easier and also helps the decision-making process. Fiahlo et al. also highlighted the importance of interoperability and scalability for the development of both physical and analytical dimensions of the system.

Khajavi et al. [35] proposed a framework for the creation of the DT of a façade through the use of IoT technologies. They started creating a WSN to collect data from the sensors located on the façade. The collected data were temperature, relative humidity and light and were eventually displayed in the visualization of the DT. In this study only a façade was examined with the use of a limited number of sensors and only three measurements were gathered.

Dave et al. (2018) proposed a platform for integrating real time data gathered by the IoT sensors and BIM, in a campus web-based system called Otaniemu3D, which gives

information about energy usage, occupancy and user comfort, through open messaging standards and Industry Foundation Classes (IFC) models. The paper indicates the design criteria, the system architecture, the workflow and a proof of concept with uses cases such as FM. In fact, the visualization of real time data is helpful for monitoring the devices and support the facility managers in decision making [43].

Natephra et al. [44] focused on visualizing live environmental data collected by IoT devices in the AR environment created based on existing models. The data, like temperature, humidity and light intensity, were collected using Arduino microcontrollers and successively stored in the BIM model and the AR application in real time. The collected data about indoor environment can also be stored in the BIM database to be further analyzed and create a historical database. The paper also confirmed the positive impact of real time data visualization for FM, in order to immediately intervene in case of malfunction and the historical database is a great resource to keep track of the components of the building life cycle.

Kazado et al. [45] proposed three approaches for integration of the building sensor technology and the BIM process so that the visualization and the analysis of real time data and historical readings is enabled. The proposed approaches are:

- Sensor- Revit-integration;
- Sensor-Revit-Naviswork-integration;
- Sensor-Revit-Naviswork-API.

This paper demonstrates how BIM-sensor integration can lead to a more responsive building maintenance and operation favoring the dialogue between stakeholders.

The first approach, that is based on the sensor-Revit integration, even if it's an easy and straightforward approach with real time data, presents several limitations. The data can be visualized only in a 2D environment and the visualization of historical data is not

permitted. Besides since the visualization happens in the software Revit, some unintentional changes to the model can occur. Eventually the lack of interoperability of Revit with other software and the fact that big file in Revit makes the program slower.

The second approach is based on the usage of the software Autodesk Naviswork Manage which permits visualization, database connectivity and appearance profiler. This approach overcomes the problems of the first approach like unintentional changes, size of the file and interoperability between other programs. But an important lack of this approach is the historical database and this represents a limitation for the understanding of the long-term behavior of the asset.

The third approach involves an add-in of Naviswork developed in VS Code using .net application programming interface (API). The sensors- Revit-Naviswork-API integration fulfill the voids of the previous approach. In fact, it enables the visualization of historical data and besides permits to visualize general information of the project and search particular asset.

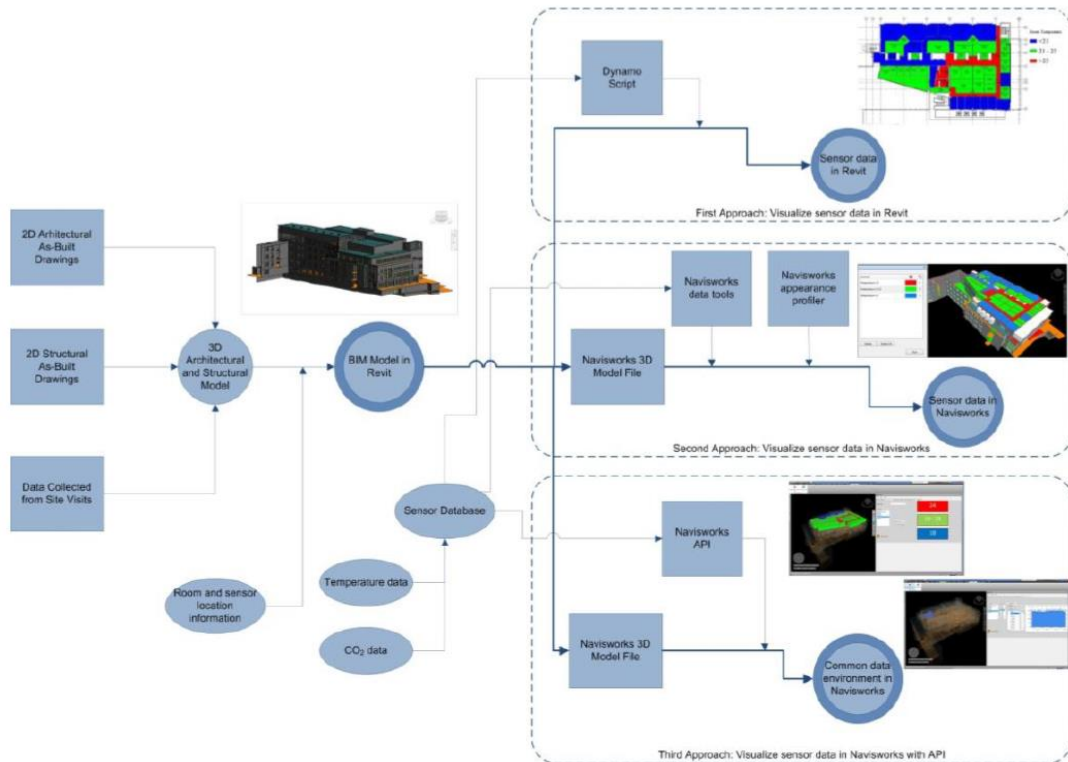


Figure 1: Research methodology process map

Figure 11 Research methodology process ap by Kazado et al.

Source: [45]

According to the study “A review of building information modelling (BIM) and the internet of things (IoT) devices integration: Present status and future trends” in the O&M phase the studies conducted can be summarized in:

- Identify physical building components and link with BIM models through RFID tags for asset tracking;
- link physical objects with digital objects by linking BMS and BIM, using BIM tool APIs for O&M;
- Extract real time data, visualize problems either from Augmented Reality (AR) devices or BIM tools on mobile device on facility managers' perspective to conduct maintenance or control of assets.

3 Cloud-based platform for building's Data Visualization

The challenge of visualizing sensors' data in building model is studied from decades. The first system that enabled data visualization was Building Management System (BMS), developed in 1980s [30]. The BMS is a system of control and management of buildings' asset, such as electrical, heating, lighting and mechanical services, safety and security [30]. However, before 2003, BMS were provided by companies which services were closed and proprietary [46]. This brought building managers to be locked-in in one of these companies, which also implied a lack of interoperability. Besides, the rising of BIM in AEC industry represented another limitation for BMS, since there is no straightforward way to integrate BMS systems with BIM, because they are incompatible and rely on different modelling approaches, languages and protocols [47].

After 2010, the advent of free access app and IoT, with all sensors that are web accessible, helped the promotion of app for building monitoring that were open and enabled interoperability.

This permitted the development of cloud-based platform for the integration of BIM and IoT.

This type of platform includes a series of advantages for FM activities. First of all, cloud-based platform reduce the cost of managing the computer infrastructure, since it would be hosted in the cloud, besides the access to data is available from anywhere since they are stored in the cloud [48]. They enable high availability of cloud computing since the cloud is composed by multiple servers and data storage units and the failure of one doesn't affect the use of the software and data [48]. For FM these systems allow the interoperability between different sources of data.

For this thesis a search was conducted in order to find a platform which could enable the integration of BIM and IoT for the creation of DT for FM usage.

3.1 Ecodomus

According to [49] Ecodomus is “a software platform that has developed from a Construction Operations Building Information Exchange (COBie) middleware to a life cycle CDE and DT platform, using a 3D model interface to integrate with multiple FM systems”.

The software purpose is to create a building DT in order to improve design and construction data collection and handover, FM, O&M. In fact, it can create, maintain and visualize BIM-based digital building twins, making design and construction data available for building O&M [50].

It is based on a CDE that integrates BIM, Building Management Systems (BMS), Computerized Maintenance Management Systems (CMMS), Geographical Information System (GIS) and IoT systems. The solution enables BIM-driven workflows and digital twin-based lifecycle management, complemented by 3D visualization [50]. Thanks to its feature it can be used as a central facility repository accessible to the stakeholders involved. The software can be used as a Software as a Service (SaaS) or through an installed on-premises. It is also available on mobile and tablet making easier the access to the platform.

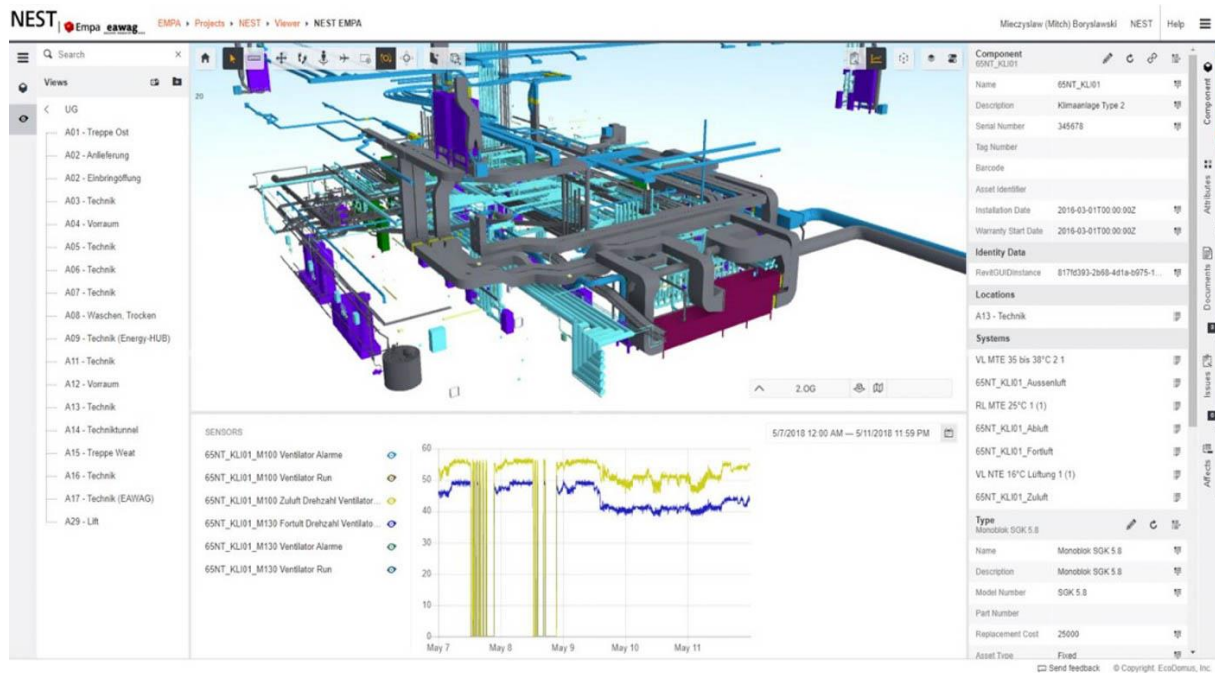


Figure 12 Ecodomus real time data interface with IoT

Source: <https://www.ebarchitects.eu/bim/7d-bim>

For this thesis there was not the possibility to test this software, even if it has the features needed for BIM-IoT integration, since the license it's not available.

3.2 Forge Autodesk

Forge, developed by the software corporation Autodesk, is a cloud development platform that includes a series of web service APIs, that permits the exchange of data between different applications. It is based on web standards such as RESTful APIs, HTML 5, CSS, and JavaScript.

The platform Forge was born in 2015 and since then has gone through many implementations and currently enables a wide variety of applications. The Forge APIs now available are wide and are described in the figure below.

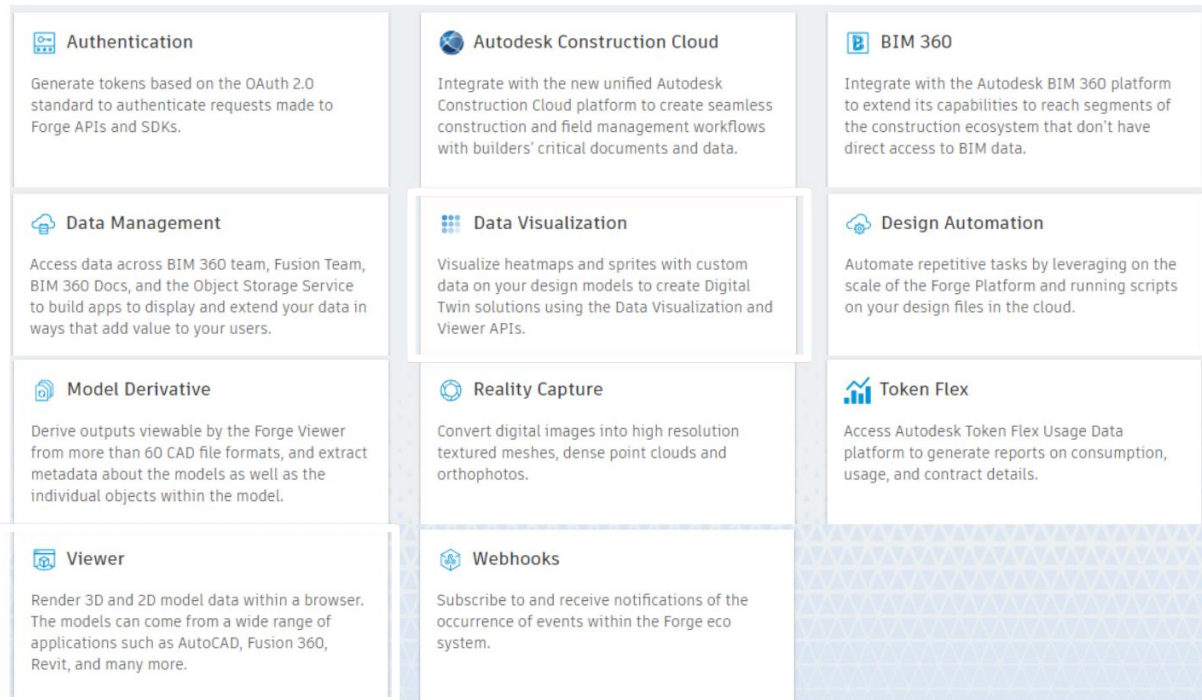


Figure 13 Forge APIs

Source: [51]

3.2.1 Data Visualization API

The Data Visualization Extension by Forge Autodesk was released in 2021 and represents an IoT toolkit to incorporate sensor data from any database service (Azure, AWS, etc.) into BIM models. In other words, it allows the visualization of the 3D model on browser implemented with sensors' data and the display of heatmaps through time.

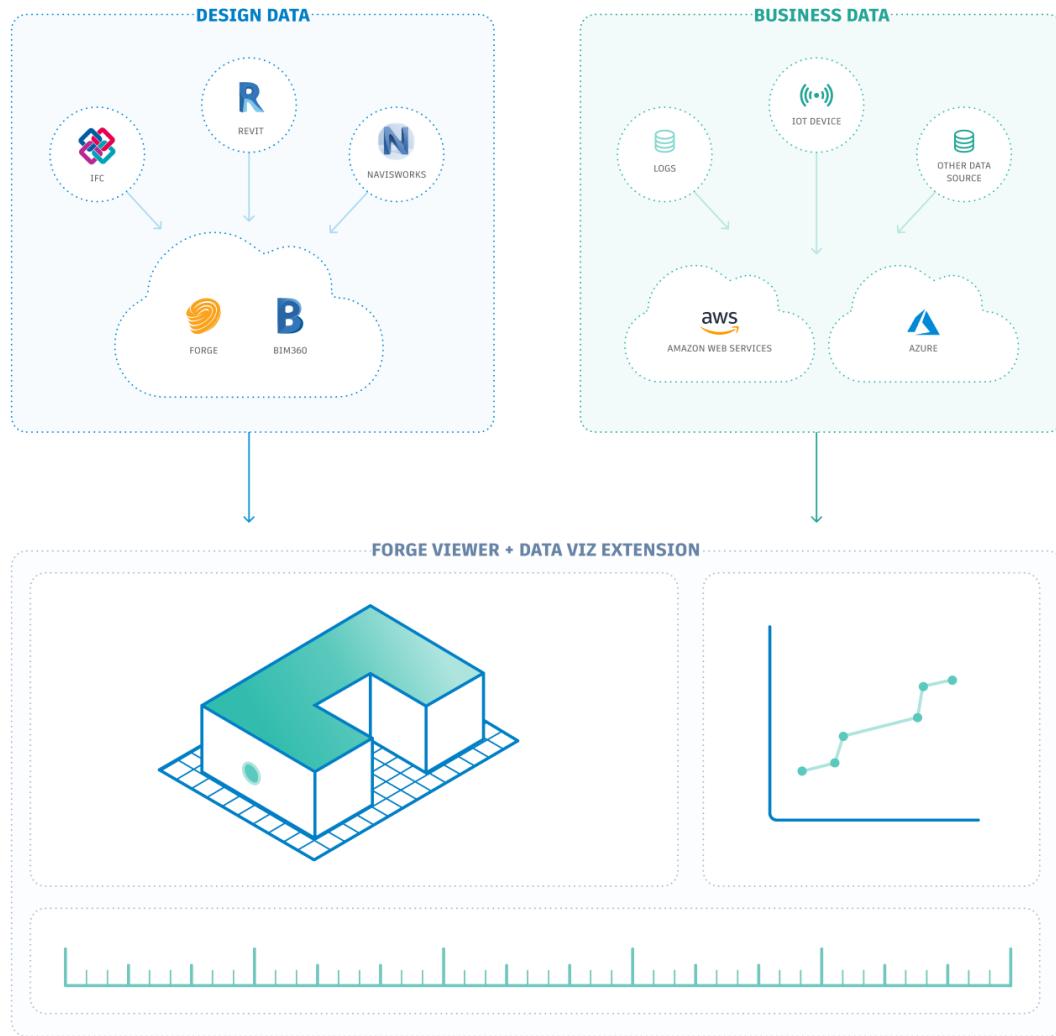


Figure 14 Workflow of Data Visualization API

Source: [52]

It is available a Reference App to explore and discover all the features of the Data Visualization Extension. In the figure below are described the Components of the Reference App.

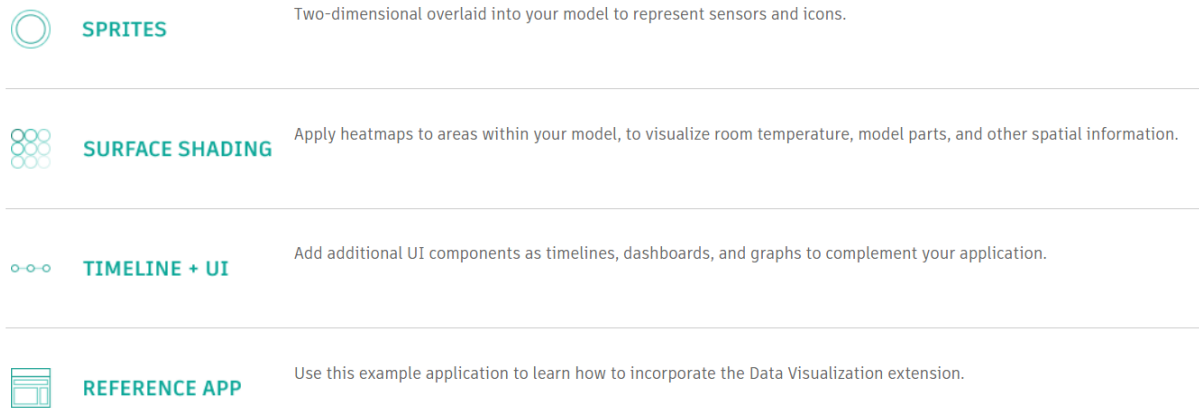


Figure 15 Components of Data Visualization API

Source: <https://forge.autodesk.com/en/docs/dataviz/v1/reference/Core/DataVisualization/>

The Data visualization app is made up of two Node Package Manager (NPM) modules (React.js client code or front-end as the User Interface (UI) framework, which is everything the user sees and interacts with, and Node.js server code or back-end which enables the interface to work) and a Reference-Application that demonstrates how they work together [51] [52].

In the figure below it's showed the viewer of Forge accessible through browser at the static address <http://localhost:9000/>. The Viewer (formerly part of the "View and Data API") is a WebGL-based, JavaScript library for 3D and 2D model rendering. 3D and 2D model data may come from a wide array of applications, such as AutoCAD, Fusion 360, Revit etc. [52]. The Viewer communicates natively with the Model Derivative API to fetch model data, complying with its authorization and security requirements. The Model Derivative API enables users to represent and share their designs in different formats, as well as to extract valuable metadata [52].

The API offers the following features [52]:

- Translate designs into SVF/SVF2 format for rendering in the Forge Viewer.
- Extract design metadata and integrate it into your app. Including object hierarchy trees, model views, and object properties, - such as materials, density and volume, etc.

- Extract selected parts of a design and export the set of geometries to the OBJ format.
- Generate different-sized thumbnails from design files.
- Translate designs into different formats, such as STL and OBJ.

Below is illustrated the process to upload and view a file in the viewer. Seed file stands for the original model file that will be visualized in the viewer and it's translated to SVF2 format and is then identified by a Uniform Resource Name (URN)

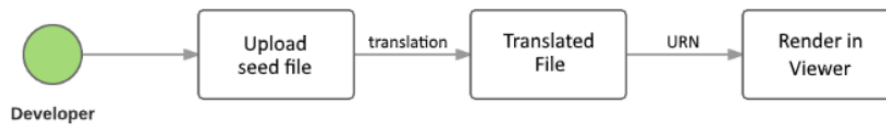


Figure 16 Process to view a model in the viewer

Source: [52]

In the next section will be described the viewer's components in detail.

In the upper part of the viewer is present the timeline, that permits to start a simulation to view the data collected according to time and to select a specific moment and view the relative data. Under the timeline is present the range of the parameter measured and the tab where it's possible to select the parameters that we want to visualize on the model. At the right of the sensor parameters and range selector is present a ViewCube, a tool to help the orientation of the camera view in the model.

In the right part of the viewer is present the sensors' list divided by floor and room. In the lower part is present a toolbar which enables to navigate through the model, to make geometrical measurements and to analyze section of the model. Besides in the right part of the toolbar is present the model browser, the command properties and the settings button.



Figure 17 Hyperion Reference App Forge viewer

1. Timeline
2. Sensor parameter and range selector
3. ViewCube
4. Layers, sensors and heatmap types selector
5. Sensor list organized by floor and rooms with graph
6. Toolbar

In the model the data are associated to sprites that represent the sensors. To a single sprite multiple parameters can be associated.

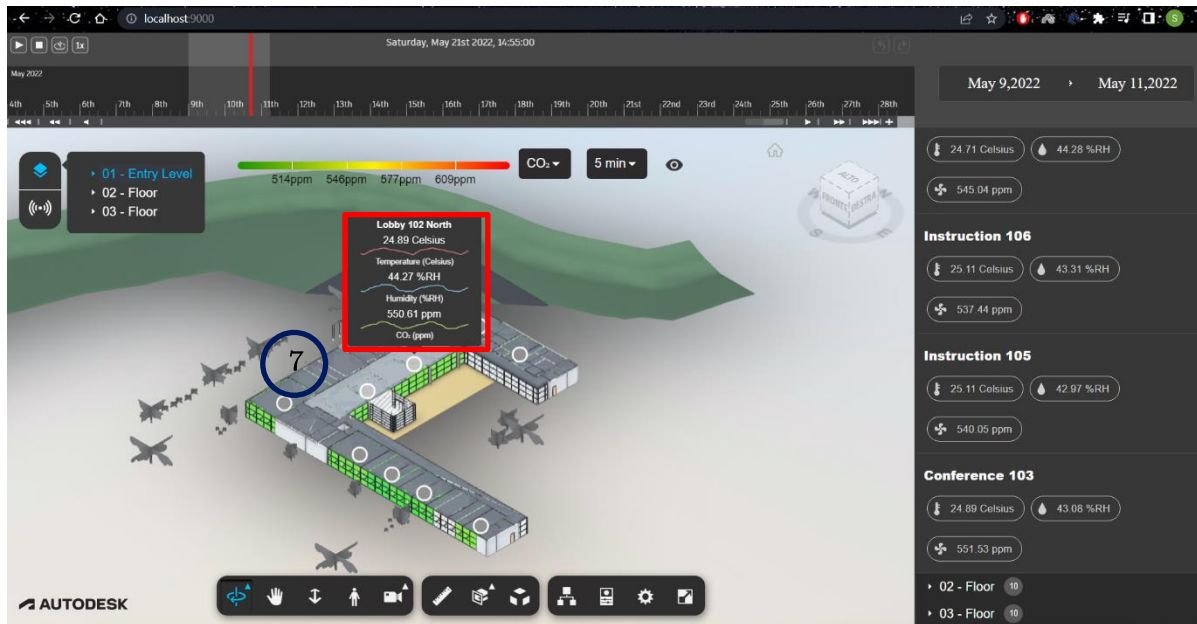


Figure 18 Hyperion App with sensors' data displayed

7. Sensors' data

Moreover in the sensors' list when a parameter is selected a graph quantity time reveals its trend according to time.

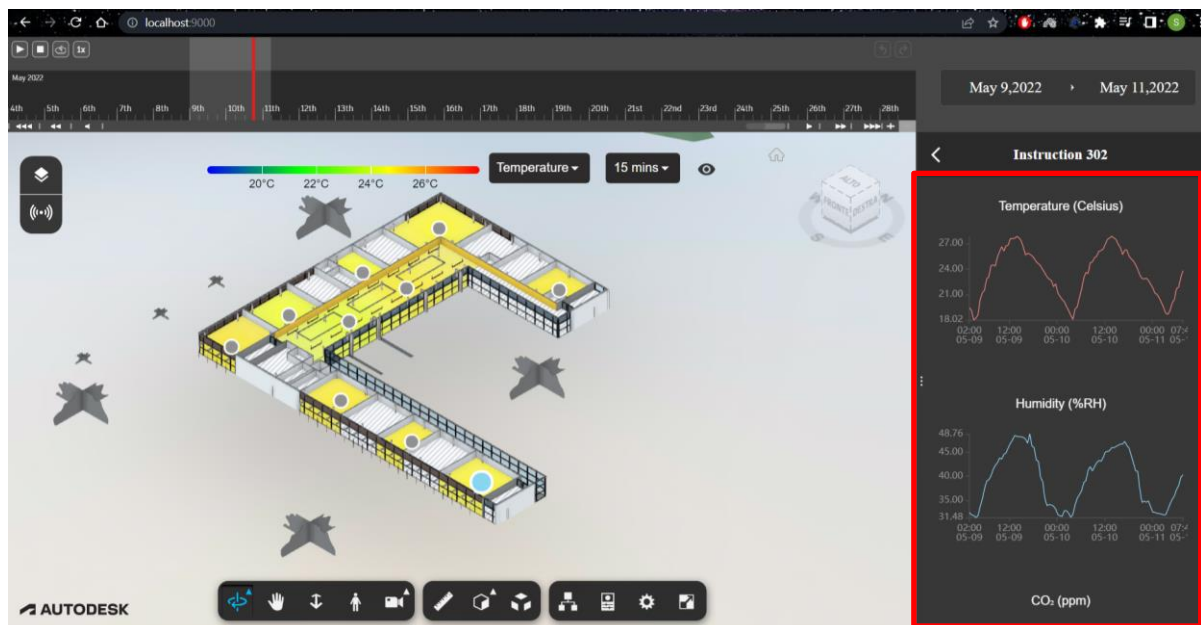


Figure 19 Graphs quantity-time of sensors' data

For this thesis this cloud-based platform was tested, thanks to availability of a 90 days free trial.

3.3 Dasher 360

Project Dasher is an Autodesk research project, born in 2009, using a BIM-based platform to provide real-time building performance throughout the life cycle of the building. It integrates sensors' data with model data from Autodesk Forge to contextualize IoT data in 3D. In fact, since the launch of Forge, the Dasher project has included some of its components, such as Data Management, Viewer, Authentication and Model Derivative and implemented them. Before the launch of Forge, it was a desktop application, while now it's a web application.

The aim of the software is to build a comprehensive framework for monitoring building performance. It can be considered as a visualization hub, where collected data from various sources is intuitively aggregated and presented in 3D [53].

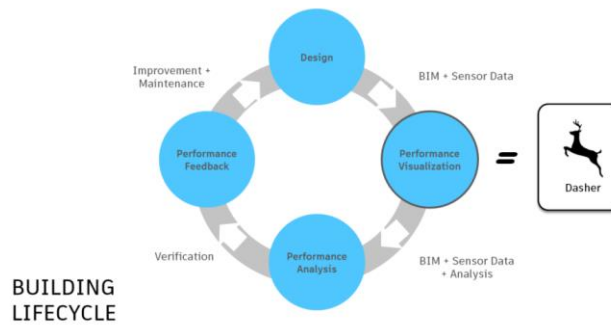


Figure 20 Dasher 360 placement in building life cycle

Source: [53]

The software enables the visualization of real time data, but also of historical data coming from the sensors placed in the building.

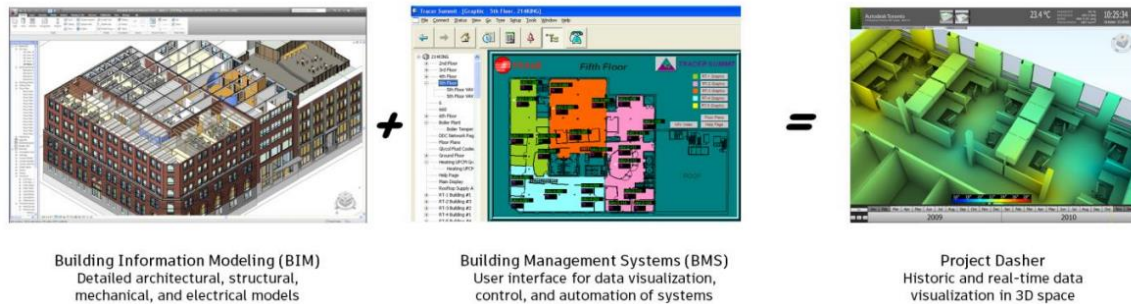


Figure 21 Concept of Dasher 360

Source: [53]

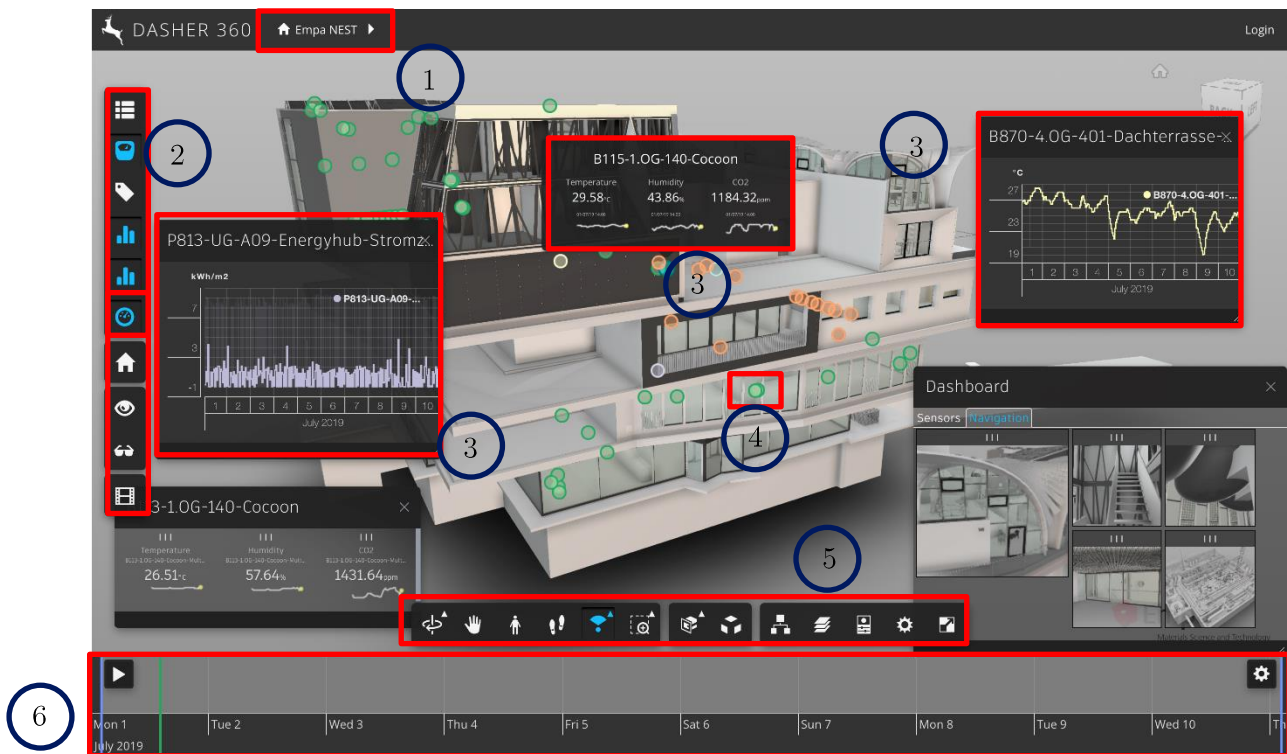


Figure 22 Dasher 360 interface with sensors' data

Source: <https://www.keanw.com/2019/09/a-major-update-to-dasher-360-now-showing-the-nest-building.html>

- ① Floor and room navigation
- ② and ⑤ Toolbar
- ③ Graph of sensors' data
- ④ Sensors
- ⑥ Timeline

Since Dasher 360 it's an ongoing research project a license is not available, so it cannot be tested in this thesis.

4 Methods

In this section will be described the methods used to achieve Data Visualization of sensors' data within a BIM model in the case study.

4.1 BIM-IoT architecture

The data acquisition workflow created is based on the implementation of the BIM-IoT architecture of the article “IoT Open-Source Architecture for the Maintenance of Building Facilities” [30]. The one present in this article is “an automated data acquisition system within the IoT and BIM integration methodology based on open-source technologies” [30].

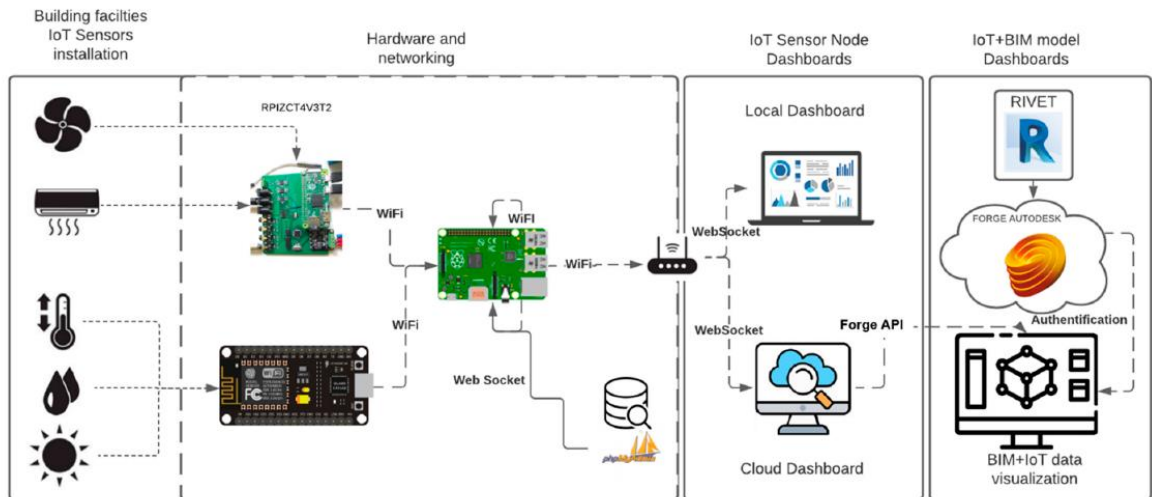


Figure 23 BIM-IoT architecture of “IoT Open-Source Architecture for the Maintenance of Building Facilities”

Source: [22]

According to [30] “all the sensors for building facilities and rooms are hosted by Arduino based microcontrollers and a Raspberry Pi 3B (Rpi3B) single board computer with integrated Wi-Fi module to acquire the planned data in real time and store them on server”. The data gathered is processed to monitor and detect anomalies in building’s assets. The BIM-IoT architecture proposed in this thesis is shown in the following figure.

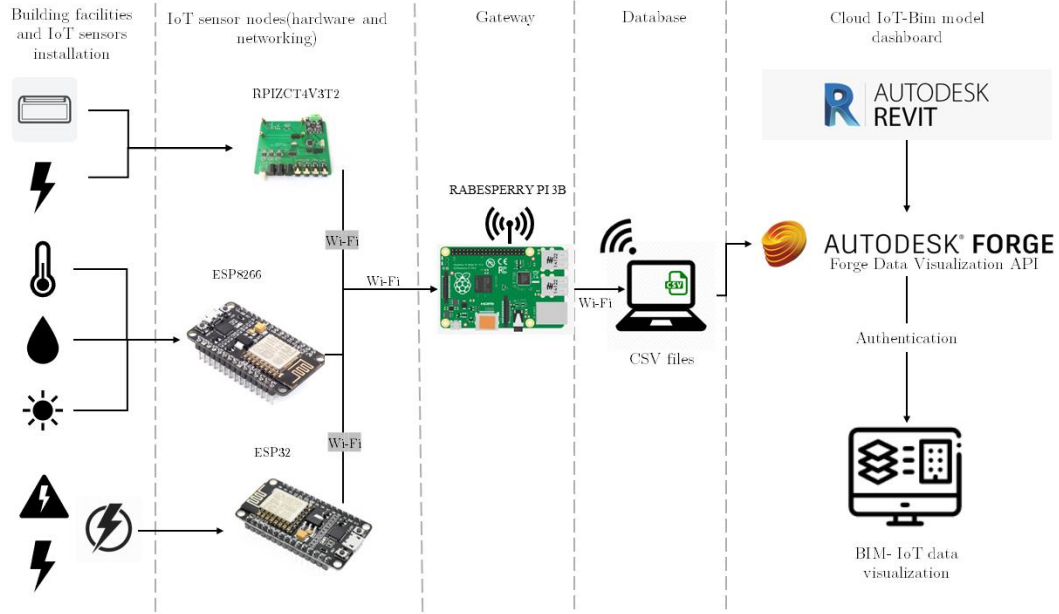


Figure 24 Implemented BIM-IoT architecture of [30]

The first column, building facilities and IoT sensors installation, represents the sensors, which collect the various data and send them to the sensors' nodes. In the hardware and networking section the raw data coming from the sensors are processed and the output gives human readable data. The next section includes the gateway, in this case the Rpi3B was used. The gateway permits the collection and the sending of detected data to the database. In this case the data were exported and collected in a CSV format. The CSV files containing the sensors' data are the database. The CSV files are eventually imported in the Forge Data Visualization API in order to be displayed within the model.

4.2 IoT system's components

The IoT system proposed is based on [30]. The system is flexible and permits the addition of new sensors and sensor nodes in real case-building facilities and to add IoT sensors data into 3D digital model of the building, according to the needs of the end-user [30]. In order to control room's internal condition and FC operation, physical parameters must be acquired. Sensors were employed at this scope and are listed in the following table.

Table 1 Sensors' specification and allocation in the building

Sensor board	Sensor name	Variable	Accuracy	Measuring ranges	Unit	Sensor allocation	
RPIZCT4V3T2	Current SCT-013-000	Irms1, Irms2, Irms3	±3	0-100 A	mA	Motor current	
	Voltage 77DE-06-09	Voltage_1,Voltage_2,Voltage_3	±5	0-230 V	Volt (V)	Motor voltage	
	Temperature DS18B20	T1	±0,5	0°– 90°C	Celsius (°C)	Delivery pipe	
	Temperature DS18B20	T2	±0,5	0°– 90°C	Celsius (°C)	Return pipe	
	Temperature DS18B20	T3	±0,5	0°– 90°C	Celsius (°C)	Air intake	
	Temperature DS18B20	T4	±0,5	0°– 90°C	Celsius (°C)	Air outlet	
	Temperature RTD(PT100)	RTD	±0,5	- 200 to 500 °C	Celsius (°C)	Motor case	
	ESP8266		Temp_ MPU	±1	-40 to 85 °C	Celsius (°C)	Motor case
		Gyroscope and accelerometer MPU6050	A-X, A-Y, A-Z	-	±250, 500, 1000 e 2000°/s	°/s	Motor case
		G-X, G-Y, G-Z	-	+2, +4 , +8 , +16 g	1 g = 9,81m/s²	Motor case	
Ambient temperature DHT22		Room temperature	±0,5	-40-80 °C	Celsius (°C)	Room	
Humidity DHT22		Room humidity	±2	0-100% RH	Relative Humidity %	Room	
	Photoresistor DHT22	Room brightness	Resistant dependen t	0-1 Ω	Lux	Room	
ESP32	Current SCT-013-000	Room current	±3	0-100 A	Ampere	Room	
	Voltage ZMPT101B	Room Voltage	±0,3%	0-1000 V	Volt (V)	Room	

Table 2 Sensors employed and embed systems

Building facilities	Sensors	Embbd systems as sensor node	Embbd IoT gateway
Room sensors	Humidity and Temperature sensor DHT22	ESP8266 development board	Raspberry Pi 3B
	Photoresistor sensor LDR		
	Gyroscope and accelerometer MPU6050		
Room sensors	Current SCT-013-000	ESP32 development board	Raspberry Pi 3B
	Voltage ZMPT101B		
	Current sensor – SCT-013-000		
Room facility	Voltage 77DE-06-09	RPIZCT4V3T2 development Board	Raspberry Pi Zero W
	Temperature DS18B20		
	Temperature RTD(PT100)		

The sensors can be used to monitor room's facilities and environmental condition of the room. In this case SCT-013-000 was used to measure the current coming to the motor and 77DE-06-09 for the relative voltage. DS18B20 sensors were used to read data about temperature of the water and the air entering and exiting the fan coil (FC).

All of these facilities' sensors are connected to the RPIZCT4V3T2 board, which is used to monitor the FC in the room.

The RTD (PT100) sensor was used to measure the motor case temperature. Also, a gyroscope and accelerometer sensor was employed, called the MPU6050, able also to monitor temperature data. The MPU6050 contains a 3-axis gyroscope and a 3-axis accelerometer. The first one measure the angular acceleration of a body on its own axis the latter measure the acceleration of a body along a direction. Its purpose is to measure vibration, motion and tilt of the FC.

The environmental conditions in the room like brightness, humidity and temperature were measured by DHT22 sensor. These, together with the MPU6050 are connected to the ESP8266 module. Eventually room current and voltage data are measured respectively by SCT-013-000 and ZMPT101B sensors, which are connected to ESP32 board. All the sensors' board are connected to the Rabeserry Pi 3B (Rpi3B) to store and display incoming data locally and remotely using wireless sensor nodes (WSN) [30].

The data about Power and Real Power are computed in the hardware level either by MCU, which is the Arduino microcontroller hosted by RPIZCT4V3T2 board, or RPIZCT4V3T2 board. The MCU function is to collect all the raw data, computing them and sending the final computation to Raspberry Pi Zero W.

The main component of the BIM-IoT system is the Rpi3B "which is a small single board computer; it operates as server, networking router, middle communicator, hosts a dashboard and a database" [30]. "The Rpi3B system connects with other sensor nodes through the Wi-Fi network and logs the received data from the sensors to a database" [30].

4.3 Fault detection methodology

The sensors installed in the FC were placed in components, whose choice was made in relation to the identification of anomalies through collected data. The **Figure 25** shows the methodology for detecting faults in the FC and planning maintenance based on the condition of the FC. It's also shown the link between possible failures coming from anomalies of the gathered data from the FC parts. According to [30] the most frequent failures are of the FC are “blocked motor, insufficient air flow, dirty filters, capacitor failure, insufficient water flow, etc.”. Each failure requires a proper action executed by maintenance operators and data collected can be used to inform FM about FC's condition and schedule preventive maintenance services.

FAN COIL	Edge machine	Maintenance		Centralized data collection			
Fault kinds	Autodecision (local)	Extemporaneous request (under condition)	Programming / Planning	For component reliability rating	To verify an anomaly in the environment	To check "zone" anomalies	Notes
False Voltage		■ (only controller, no fan coil)		■ (only controller, no fan coil)		■ (only controller, no fan coil)	
Motor blocked		■		■		■	
Operating hours / exhaustion filters			■ cumulative (filter cleaning, bearing lubrication if necessary)		■	■	the environmental anomaly is signaled when an environment is different from a "similar" activation period of more than "X" time the average(deleted sample extreme values). The "zone" anomaly is signaled when a large part of the area of the fan presents a high number of hours in relation to the predictable(as a function of ambient data and user programming)
Unbalancing of rotating masses				■			
Excessive resistant torque (motor overheating)	■ limit value exceeded	■ exceeded recommended value		■		■	the anomaly is signaled if most of the fan coils in the area show overheating
Insufficient water circulation		■				■	the "zone" anomaly is signaled when most of the fan coils in the zone have insufficient water flow
Delivery temperature anomaly (water)		■			■	■	the room anomaly is signaled when a single fan coil does not receive water supply(delivery temp. = return temp.) during the service activation period. The "zone" anomaly is signaled when a large part of the fan coils of the area is supplied with water temperature that does not correspond to that provided by the management system
Insufficient air circulation		■		■			

Figure 25 Kinds of faults and data collection for preventive maintenance of the FC

Source: [30]

The **Figure 26** shows the algorithm and alarm system implemented on the RPIZCT4V3T2 sensor board. It can be divided in three main sections:

- Installed sensors in the FC;
- Management and monitoring systems of the FC components;
- Alarming FMs or end-user when anomalies occur.

The algorithm's decisions are based on sensors' values compared to the installation conditions.

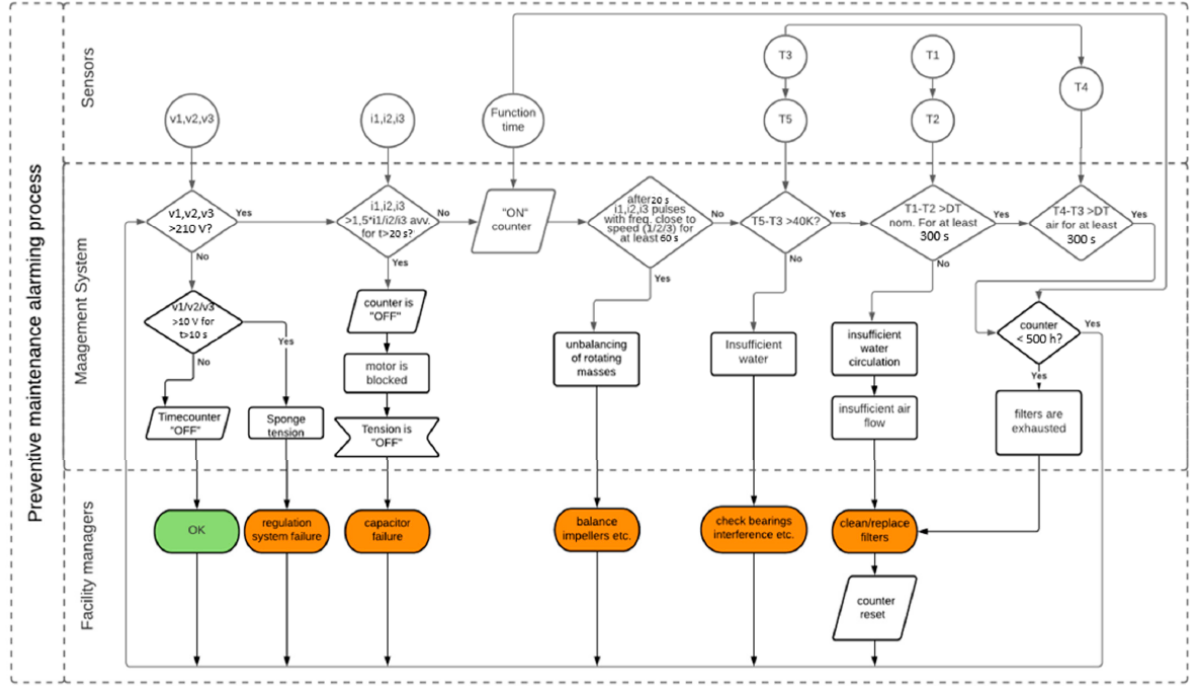


Figure 26 Integrated anomaly detection flowchart

Source: [30]

It is also important to define the time range in which the measurements are taken. The **Table 3** describes the sampling frequency for FC's components depending on the power condition. The motor temperature must be monitored every 10 s and send average values every 30 minutes to the server [30]. The motor voltage must be monitored more often, besides the electric power should be monitored only in the first 10 s, when the voltage is above 200 V. Instead, the temperature of the air and the water entering and exiting the FC must be monitored every 30 minutes.

Table 3 Sampling frequency and FC sensor allocation Source: [30]

Sensors	Frequency	Sensor Allocation	Note
T1	1800"	delivery pipe	every 1800" send to the server the average detected values in the interval in which one of the values is at least 200 volts
T2	1800"	return pipe	every 1800" send to the server the average detected values in the interval in which one of the values is at least 200 volts
T3	1800"	air inlet	every 1800" send to the server the average detected values in the interval in which one of the values is at least 200 volts
T4	1800"	air outlet	every 1800" send to the server the average detected values in the interval in which one of the values is at least 200 volts
T5	10"	motor case	every 1800" send to the server the average detected values in the interval in which one of the values is at least 200 volts
v1	0.1"	motor voltage	send voltage value to the server only if for at least 180" v1 is greater than 0 and less than 200 volts
v2	0.1"	motor voltage	send voltage value to the server only if for at least 180" v1 is greater than 0 and less than 200 volts
v3	0.1"	motor voltage	send voltage value to the server only if for at least 180" v1 is greater than 0 and less than 200 volts
i1	0.05"/3"	motor current	0.05" for the first 10' from v1 equal to at least 200 volts (anti-unbalance)/3" in normal operation/no fields-on if v1 < 200 volts/send to the server the integral of $i1 \times dt$ on the range where v1 is at least 200 volts (power consumption control)/send to alarm server if $i1 > 1.5 \times i1$ rated for more than 6"
i2	0.05"/3"	motor current	0.05" for the first 10' from v2 equal to at least 200 volts (anti-unbalance)/3" in normal operation/no fields-on if v2 < 200 volts/send to the server the integral of $i1 \times dt$ on the range where v2 is at least 200 volts (power consumption control)/send to alarm server if $i2 > 1.5 \times i2$ rated for more than 6"
i3	0.05"/3"	motor current	0.05" for the first 10' from v3 equal to at least 200 volts (anti-unbalance)/3" in normal operation/no fields-on if v3 < 200 volts/send to the server the integral of $i3 \times dt$ on the range where v3 is at least 200 volts (power consumption control)/send to alarm server if $i3 > 1.5 \times i1$ rated for more than 6"

4.4 Data transmission and visualization

In this case the integration of IoT technologies with BIM happens on the Forge platform (Chapter 3.2 Forge Autodesk). The 3D model of DISEG was modeled on the 2022 version of Revit and was imported in the Forge App.

BIM-IoT integration and the visualization process using Forge is displayed in the following figure [30]. In the right column are present the services used for the custom application on Forge platform, while the left column shows the applications layer [30]. The middle column represents the browser visualization on the Forge viewer, which can be modified with addition of buttons, extensions or plots using the JavaScript programming language that supports the Open-source VS Code editor [30].

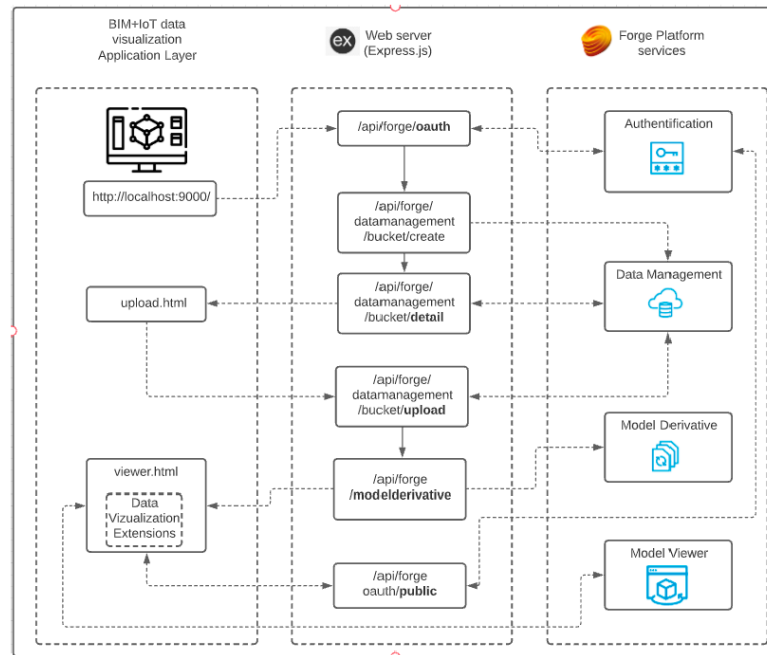


Figure 27 BIM-IoT data integration and visualization approach diagram.

Source: [30]

5 Case study

The case study is the department of structural geotechnical and building engineering (DISEG), located in Turin at the Polytechnic of Turin. The department was modelled in 3D using the software Revit 2022. All department's room are provided of FC, positioned under the windows. The FC object of study is produced by EuroMotors Italia and is type FC83M-2014/1 with 4 possible speeds. Its motor rotates in an anti-clockwise direction at 1100 revolutions per minute (RPM) at maximum speed. The FC is provided of cooling and heating battery and a filter that must be monitored. So that predictive maintenance is enabled sensors are installed to gather real time data.

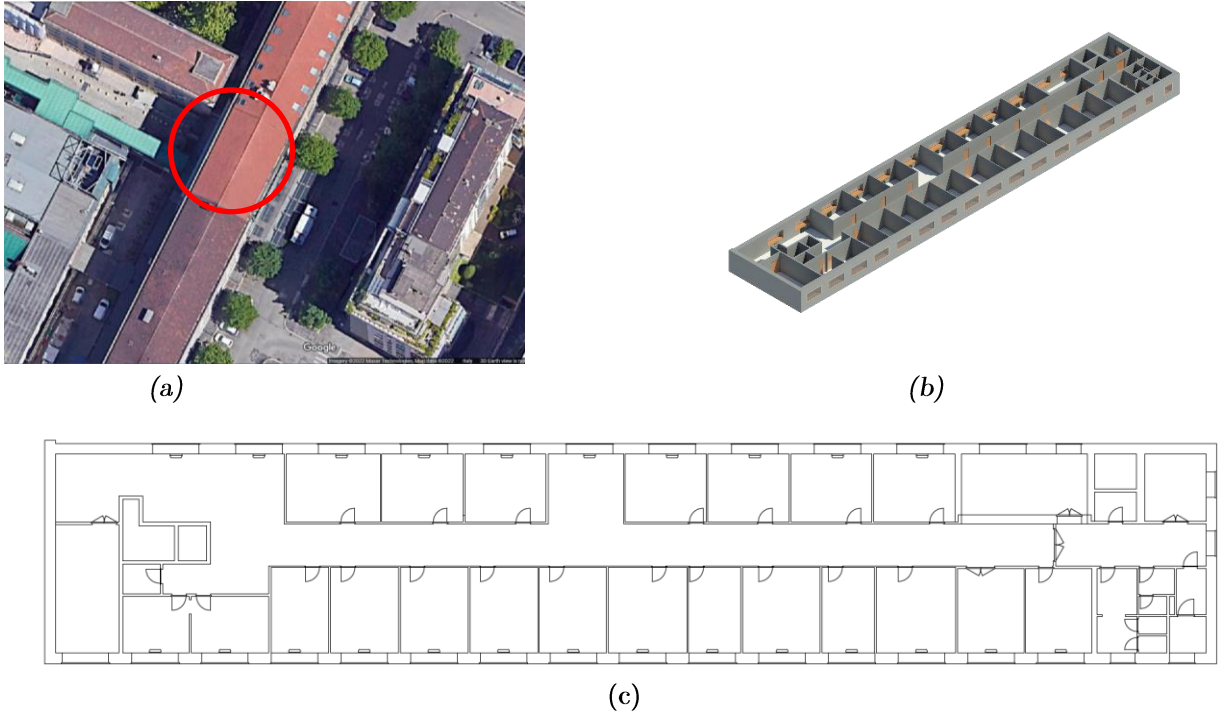


Figure 28 Case study: (a) location, (b) 3D view of the DISEG department on Revit 2022, (c) plan view of DISEG department on Revit 2022

Table 4 FC features Source: [30]

Model Number	V/HZ	Ampere	Num Speeds	Power	RPM
FC 83M-2014/1	230–240V/50	0.23 A	4	14/53 W	1100

5.1 Data acquisition

The data acquisition with IoT architecture is described in chapter 3.2 4.2 IoT system's components.

5.2 Data integration

5.2.1 Set up of Data Visualization Forge APIs

The data visualization APIs permits real time or historical visualization of sensor data in the BIM model.

For this thesis was used the 90 days free trial of Forge, which includes free access to all Forge APIs, 100 cloud credit and 5 GB of storage and services such as authentication (two-legged authentication), data management, Model Derivative, Model Viewer, all of them necessary and sufficient to create the customized application. The first step was to create or login with an Autodesk account into the Forge portal.

After getting access to the Forge platform, the next procedure was creating an app to have a Client ID and a Client Secret.

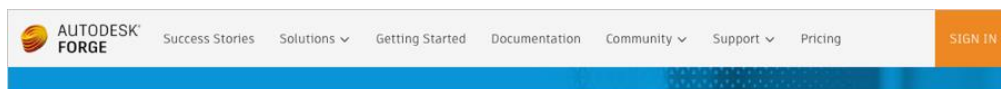


Figure 29 Forge Portal

Source: <https://forge.autodesk.com/en/docs/oauth/v2/tutorials/create-app/>

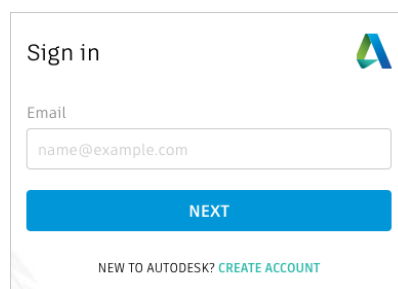


Figure 30 Sign in page on Forge portal

Source: <https://forge.autodesk.com/en/docs/oauth/v2/tutorials/create-app/>

After signing in in the Forge Portal it is possible to create an app, by clicking on “My Apps” from the profile menu.

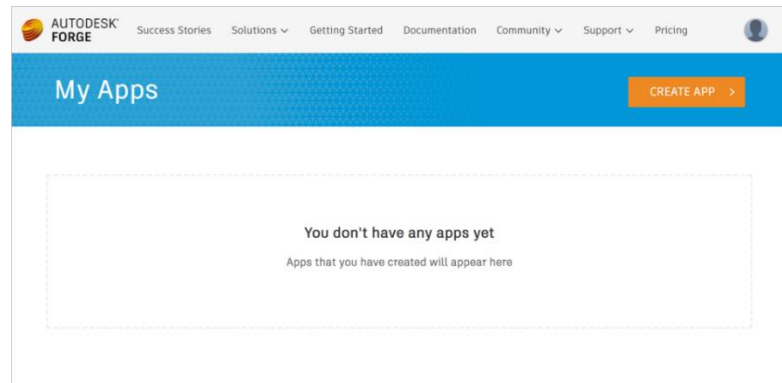


Figure 31 My apps page on Forge portal

Source: <https://forge.autodesk.com/en/docs/oauth/v2/tutorials/create-app/>

The next step is to select the APIs enabled in the App

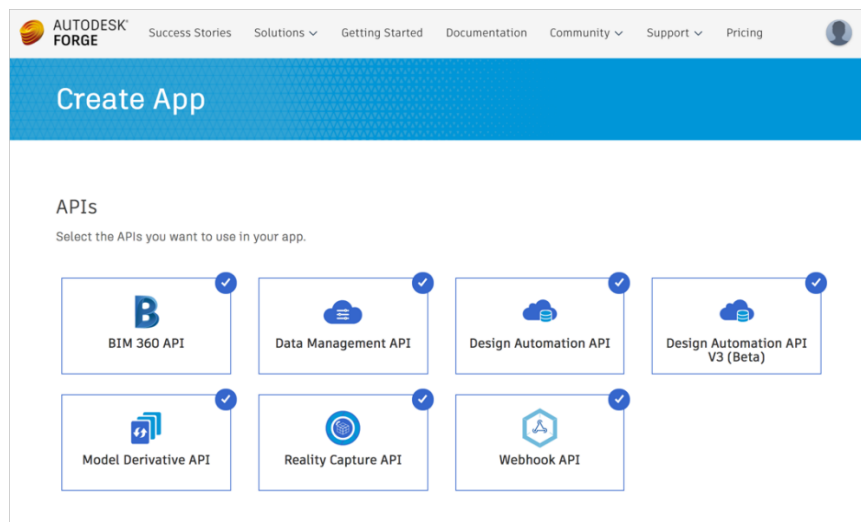


Figure 32 Selection of the API enabled in the App

The last step is to indicate the name app, giving a brief description of the application, a Callback URL, which is a URL that Forge will use to redirect the end user after acquiring authorization for the app.



App information (Created 26 Apr 2022)

About this app

App Name
Data visualization BIM-IoT

Client ID
R5JGVGLsGaGvjQUaijGAXx96CgPrzsPH

Client Secret
[REDACTED] [EYE] [REGENERATE]

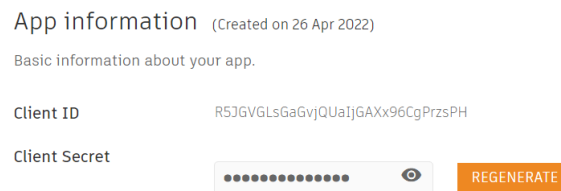
App description
[Bim-iot sensors' data integration](#)

Callback URL [What is this ?](#)
<http://localhost:8123/vscode-forge-tools/auth/callback>

Your Website URL

Figure 33 Input of app's name and description and Callback URL

The app is created the Client ID and Client Secret are provided.



App information (Created on 26 Apr 2022)

Basic information about your app.

Client ID
R5JGVGLsGaGvjQUaijGAXx96CgPrzsPH

Client Secret
[REDACTED] [EYE] [REGENERATE]

Figure 34 Client ID and Client Secret of Forge App

After getting the credentials to access the Forge App, the next step is to install and run the Reference Application.

5.2.2 Running the Reference Application

Successively was installed and run the Reference Application to display a static web page showing the Forge Viewer, which displays sample data rendered on a 3D model.

The integrated development environment (IDE) used was Visual Studio Code (VS Code) version 1.67.2 and the language chosen was Node.js, but also .NET Framework, .NET Core, Go, PHP and Java were available. Since the usage of Node.js language, it was necessary the installation of Node.js version 16.14.2 for 64-bit systems. It is an open-source cross-platform Javascript runtime environment [48] and it includes two NPM modules (React UI components and Client Server Data Module Components) that are the packet manager of Node.js.

It is important to run VS Code as an administrator to execute all the commands needed without errors. In the computer where Forge application will run, it is also important to install the Chocolatey software, which is a machine-level, command-line package manager and installer for Windows software, which simplifies downloading and installing software. Also, Python compiler was installed on the computer for the correct running of the application.

To install the Reference Application, the tutorial present on the Forge website has been followed. The first step was to clone the GitHub repository (which is a central place in which an aggregation of data is kept and maintained in an organized way, in this case in GitHub that is a code hosting platform), with the following command in the terminal of VS Code:

```
"git clone https://github.com/Autodesk-Forge/forge-dataviz-iot-reference-app.git"
```

To run this command, it was necessary the installation of the Git software for Windows. In this case the version 2.35.3 was installed.

Since the Node.js version used was 16.14.12, it was necessary to upgrade the node-sass dependency from "5.0.0" to "6.0.1" in the package.json file, also because it was an outdated node-sass package.

After cloning the GitHub repository, it is necessary to execute the command "cd forge-dataviz-iot-reference-app" to enter in the folder project and then type "npm install" to install all the dependencies. After all this steps, the application is available at the static address <http://localhost:9000> with the default model.

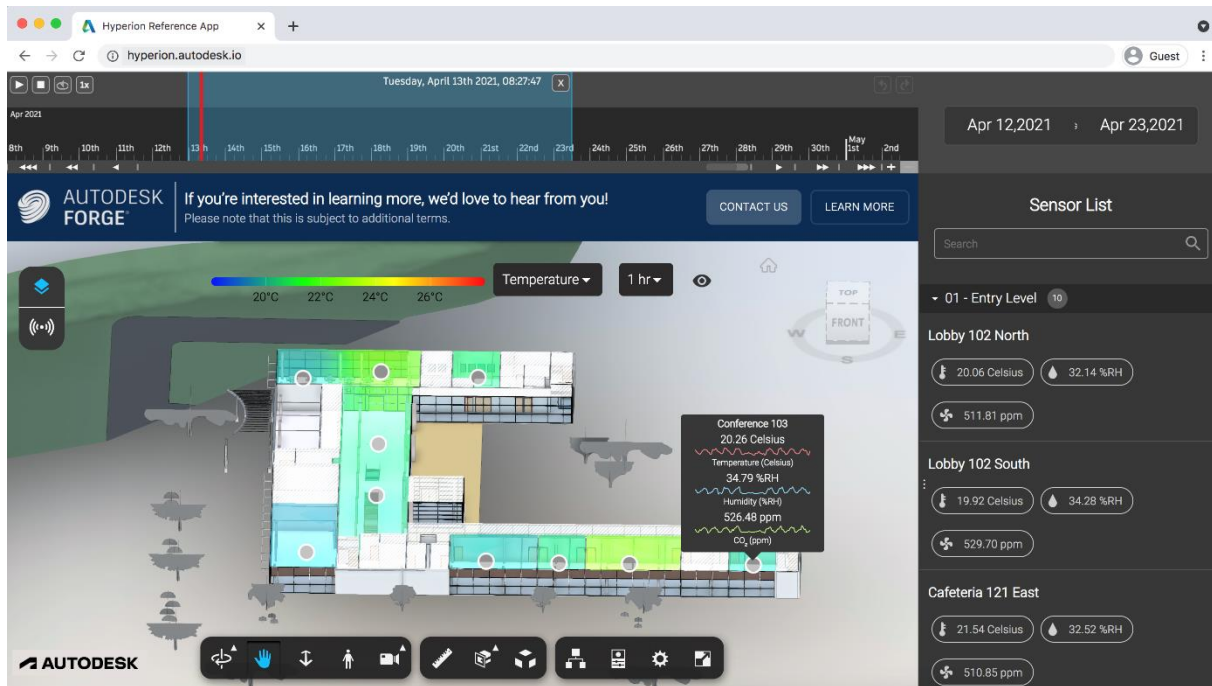


Figure 35 Forge app with the default model at <http://localhost:9000> static address

Source: [52]

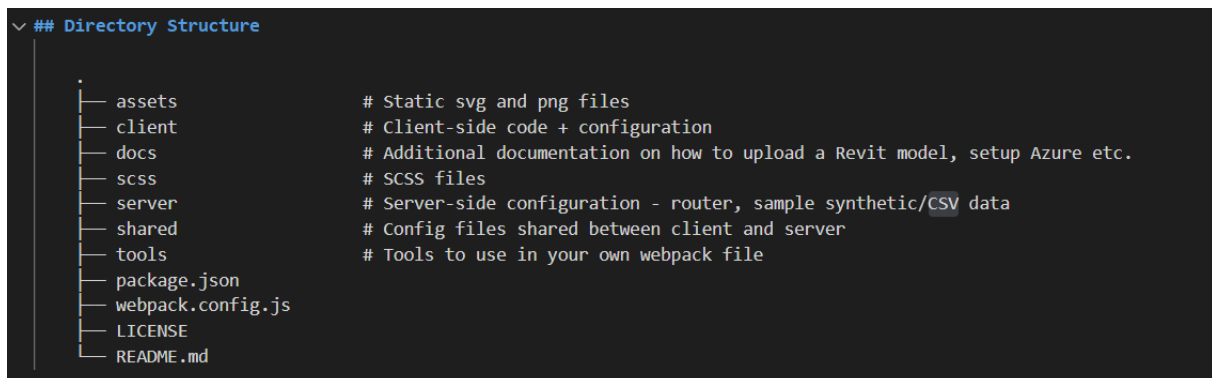


Figure 36 Reference application directory structure

5.2.3 Replacing the default model

It's now necessary to replace the default model with the DISEG's model, which was created with the 2022 English version of Revit, which is the recommended version since there could be problems with the translation of some elements, like "rooms", with the previous versions.

The Reference Application has UI features that take advantage of Rooms and Floors information contained within the metadata of Revit or IFC files, but these must be correctly set up in the Revit model. Because of this during the modelling, in order to

display, for example, heatmaps associated to each room, it was essential the creation of room tag.

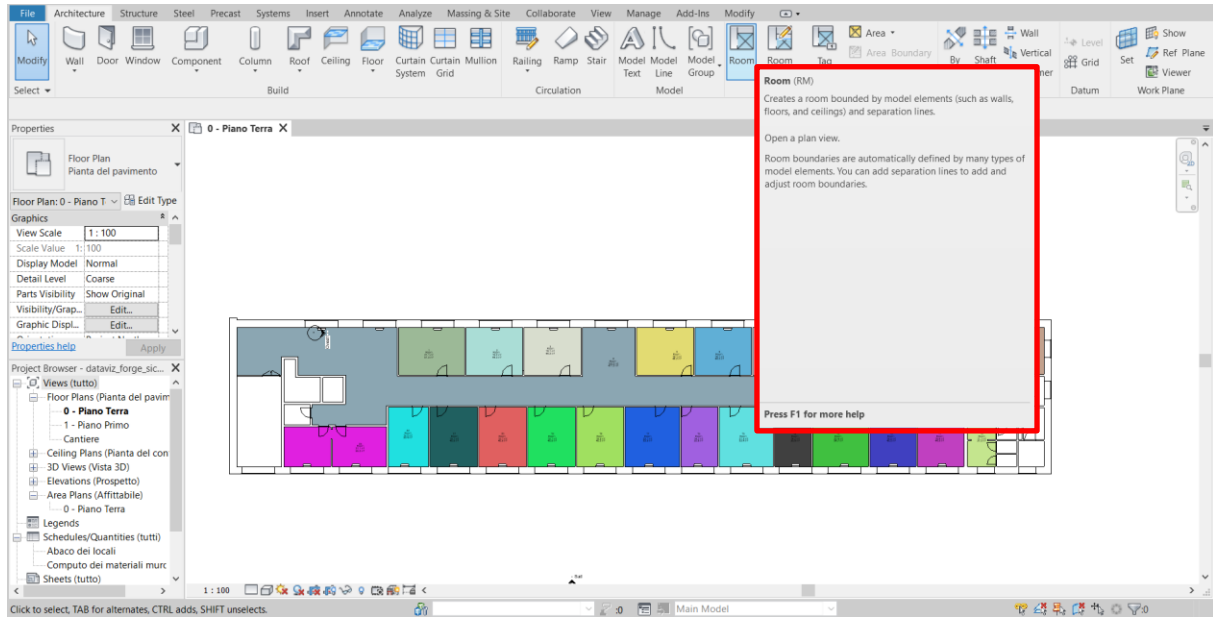


Figure 37 Creation of rooms in Revit 2022

In this case the model was firstly translated and uploaded in Forge by the means of the VS Code extension of Forge. Thanks to the extension the .rvt file can be uploaded directly from the IDE. Forge extension permits the creation of buckets, the upload of the 3D model and the relative translation in SVF2 format, needed to view the model in the browser.

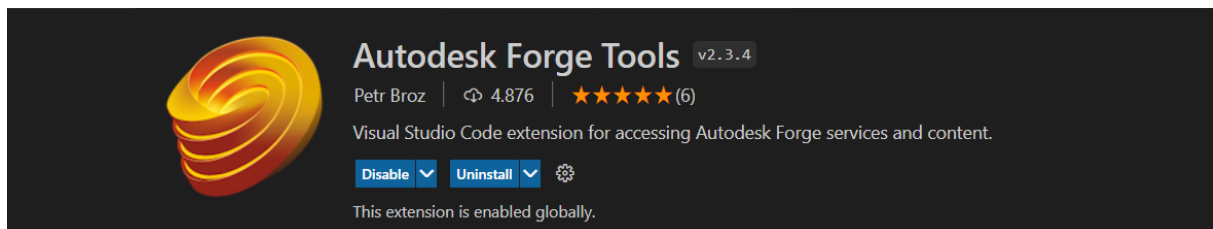


Figure 38 Forge extension for VS Code

In order to use the Forge extension it's necessary to indicate in the setting.json file, present in the settings of the extension, the "FORGE_CLIENT_ID", which is the client id of the Forge App previously created, "FORGE_CLIENT_SECRET_ID", which and the region where the extension is used, in this case Europe so EMEA was indicated.

```

C: > Users > Stefania Sic > AppData > Roaming > Code > User > settings.json > ...
1  {
2      "http.proxyAuthorization": null,
3      "autodesk.forge.environments": [
4
5
6
7
8          {
9              "title": "my env",
10             "clientId": "R5JGVGLsGaGvjQUaIjGAXx96CgPrzsPH",
11             "clientSecret": [REDACTED],
12             "region": "EMEA"
13         }
14     ],
15     "json.maxItemsComputed": 20000
16 }

```

Figure 39 Insertion of *FORGE_CLIENT_ID*, *FORGE_CLIENT_SECRET* and region on *setting.json* in the VS Code Forge extension

Thereafter is required 3-legged authentication and the extension is ready to use.

In the figure below is displayed the configuration of the Forge Extension in VS Code highlighting the buckets and derivatives section where the models are stored in SVF2 format.

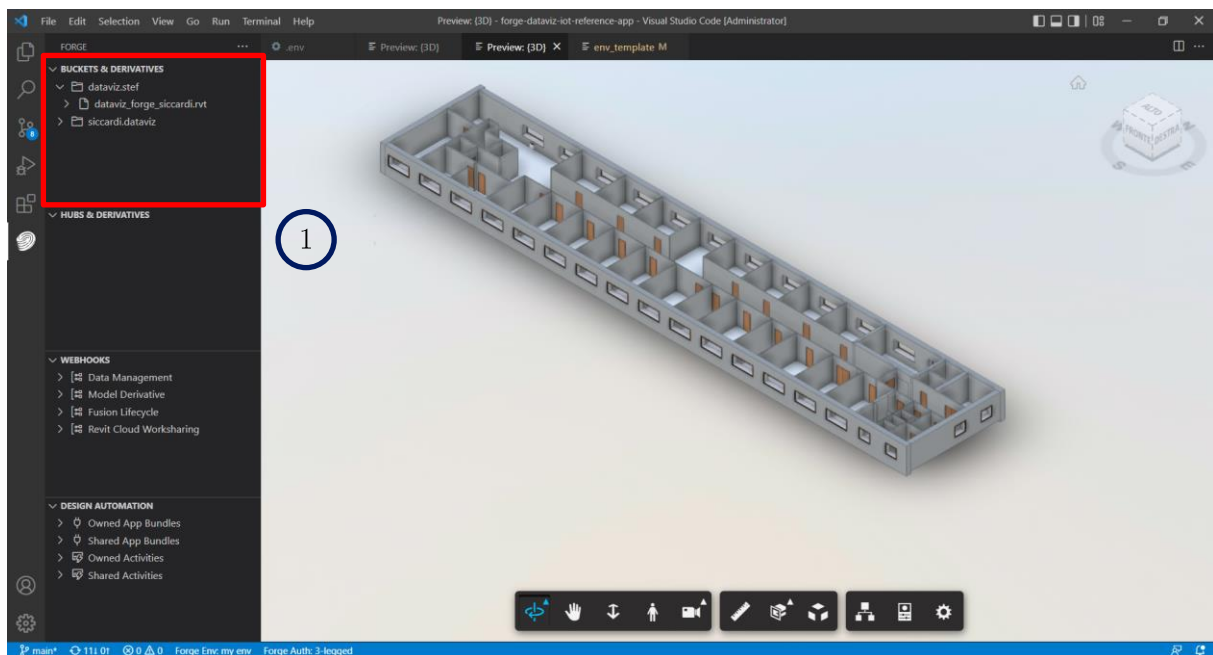


Figure 40 Viewer in the IDE through Forge extension for VS Code with the DISEG model

- 1 Bucket and derivatives section, where bucket are created managed and model can be visualized

The bucket are arbitrary spaces that are created by applications and are used to store objects for later retrieval. Each bucket has a retention policy and the retention period that specifies how long the data is to be kept. In forge the retention period can be:

- Transient, the element is stored for 24 hours;
- Temporary, the element is stored for 30 days;
- Persistent the element is stored until it's deleted.

The FORGE_BUCKET variable must be globally unique to be created.

Once the chosen model is uploaded and translated in SVF2 format, an URN is created, which uniquely identifies the model uploaded.

Before launching the application and view the uploaded model at the static address <http://localhost:9000>, the environmental variables need to be defined. In this case these were set up directly from the Windows system, since the method on Developer's Guide on Forge site was not working.

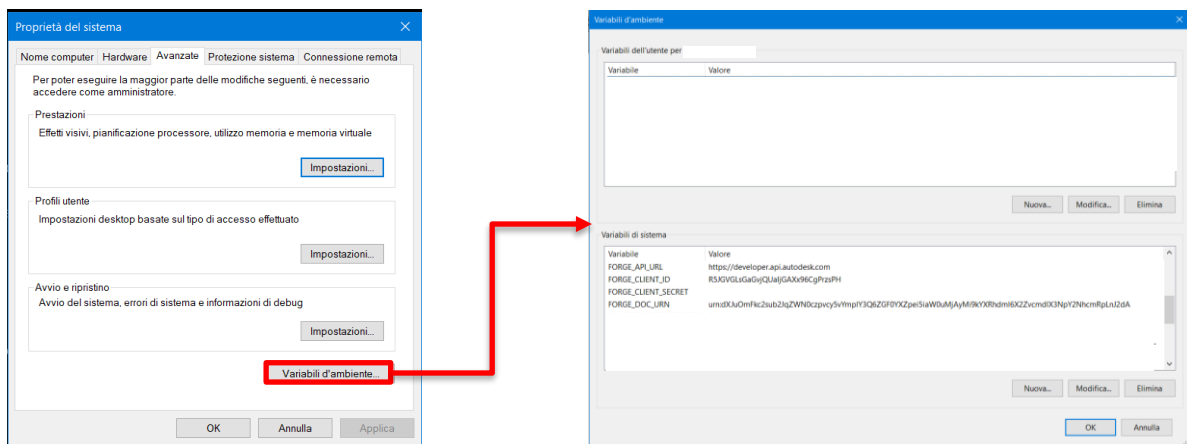


Figure 41 Set up of environmental variables from advanced setting on windows

The following environmental variables were set for the visualization of the uploaded model on <http://localhost:9000>:

- FORGE_CLIENT_ID, which is the client id of the Forge app created;
- FORGE_CLIENT_SECRET which is the secret id of the Forge app created;
- FORGE_DOC_URN, which uniquely identifies the uploaded model;
- FORGE_API_URL.

The FORGE_DOC_URN corresponds to the URN code with the prefix “urn:”. It is available for the model uploaded through the Forge extension of VS Code by right clicking on the .rvt file uploaded in the bucket and selecting “View Object details” at Object ID.

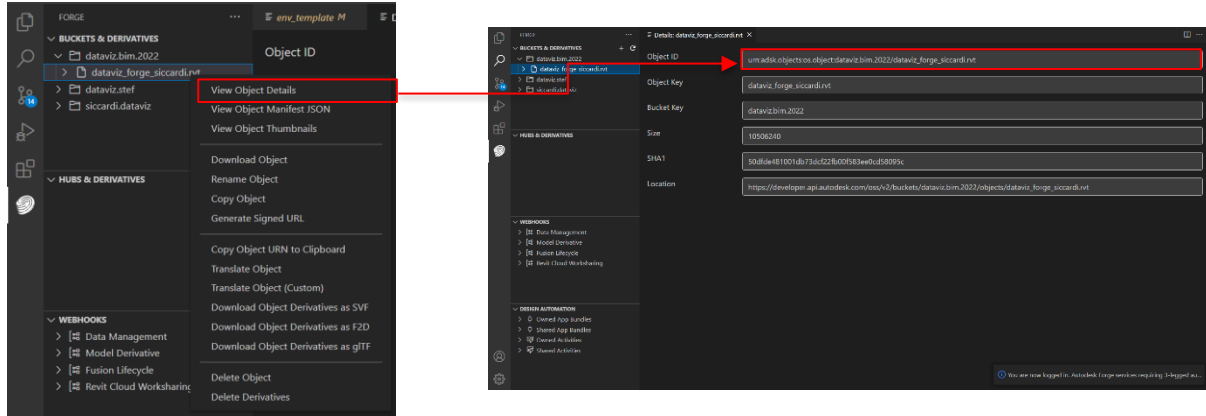


Figure 42 URN object

The 3D model can also be uploaded by creating an .env in the server folder of the project and here set the environmental variables (FORGE_CLIENT_ID, FORGE_CLIENT_SECRET and FORGE_BUCKET).



Figure 43 Setting of environmental variables on .env file

Successively the ENV variable needs to be set to the string local and the application can be launched and at localhost:9000/upload the model can be uploaded.

Select a model to upload and translate: No file selected.
☐ Enable SVF2 Upload

Figure 44 <http://localhost:9000/upload>

Once the model is uploaded the figure below displays the viewer with the new model available on <http://localhost:9000>.

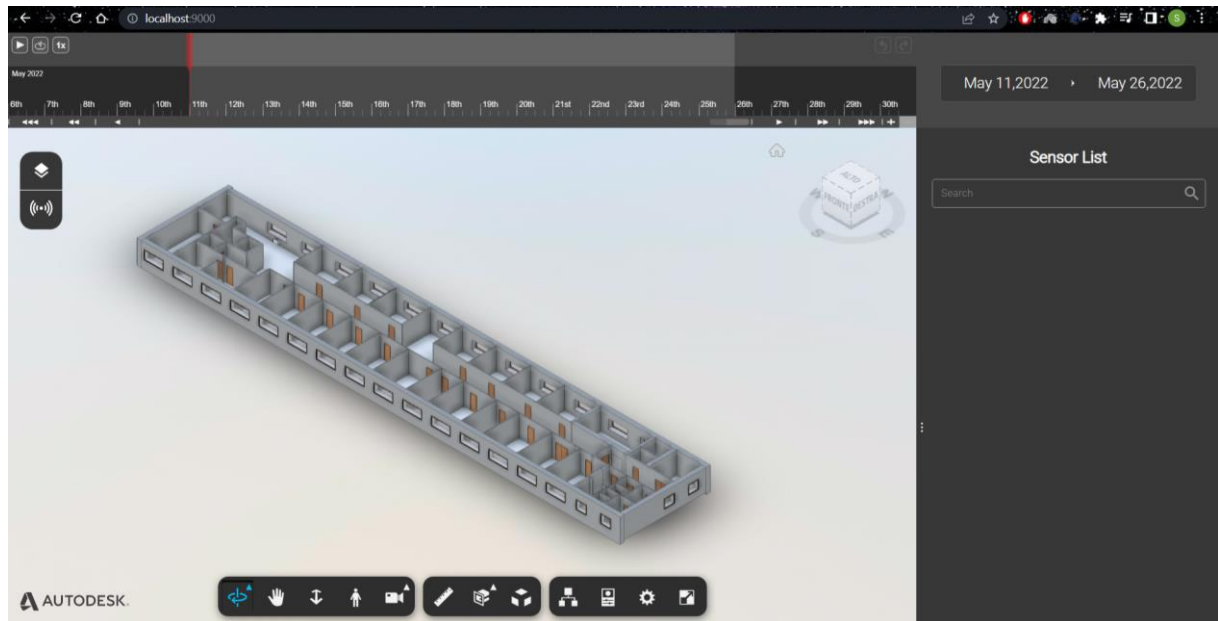


Figure 45 DISEG's model displayed in the Forge Viewer on the static address <http://localhost:9000>

It is important that the translation of the model by the Model Derivative API includes the “Rooms”, which are necessary to create heatmaps associated to each sensor placed in the model.

```

149 forge-dataviz-iot-reference-app > client > pages > JS DataHelper.js > DataHelper > createShadingGroupByFloor > getShadingData
150   async createShadingGroupByFloor (viewer, model, deviceList, nodeName) {
151     let dataVizExt = await this.getExtension(viewer);
152     const getShadingData = async () => {
153       const structureInfo = new Autodesk.DataVisualization.Core.
154       /**
155        * We have a custom type here for the UI in {@link constr
156        */
157       let shadingData = await structureInfo.generateSurfaceShadi
158         deviceList,
159         undefined,
160         nodeName
161     });
162   };
163   return shadingData;
164 };
165
166 // Ensure that instance tree has loaded.
167 await dataVizExt.waitForInstanceTree(model);
168 return getShadingData();
169 }
170
171 /**
172  * Constructs a device tree used to load the device UI.
173  *
174  * @param {SurfaceShadingData} shadingData The `SurfaceShadingData`
175  * the output of {@link createShadingGroupByFloor} or {@link creat
176
149   async createShadingGroupByFloor (viewer, model, deviceList, nodeName) {
150     let dataVizExt = await this.getExtension(viewer);
151     const getShadingData = async () => {
152       const structureInfo = new Autodesk.DataVisualization.Core.
153       /**
154        * We have a custom type here for the UI in {@link constr
155        */
156       let shadingData = await structureInfo.generateSurfaceShadi
157         deviceList,
158         undefined,
159         "Rooms"
160     });
161   };
162   return shadingData;
163 };
164
165 // Ensure that instance tree has loaded.
166 await dataVizExt.waitForInstanceTree(model);
167 return getShadingData();
168 }
169
170 /**
171  * Constructs a device tree used to load the device UI.
172  *
173  * @param {SurfaceShadingData} shadingData The `SurfaceShadingData`
174  * the output of {@link createShadingGroupByFloor} or {@link creat
175

```

Figure 46 Modified code to associate heatmaps to rooms

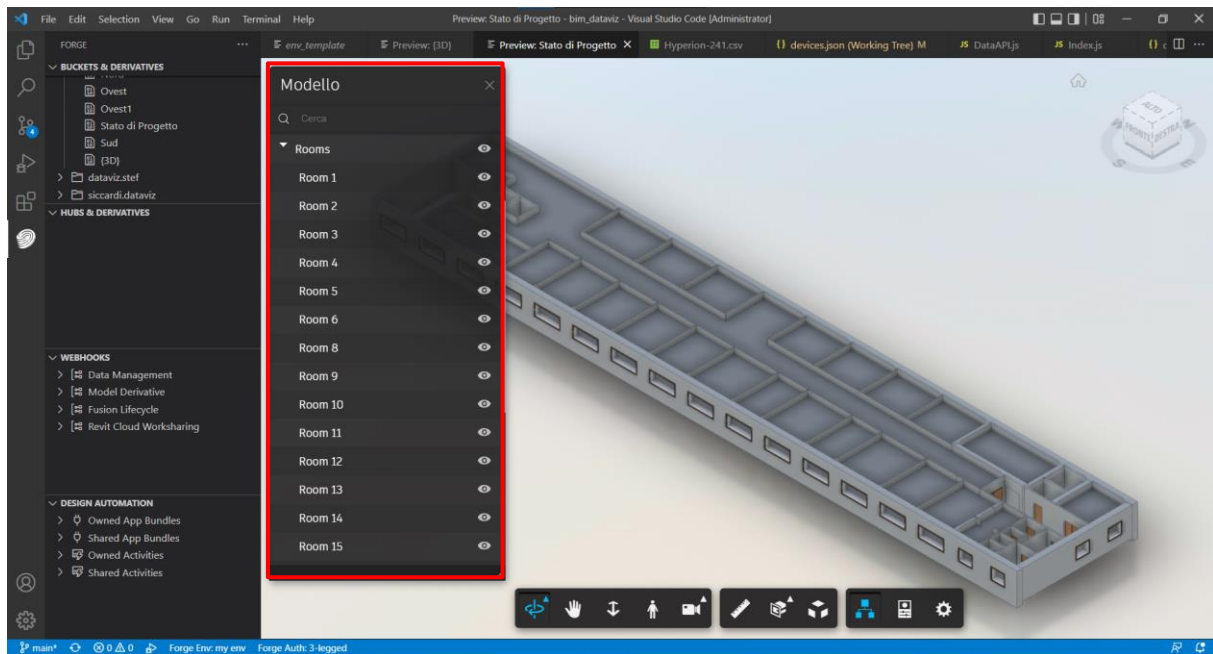


Figure 47 Rooms elements in Forge extension of VS Code

5.2.4 Adding our own sensor data on the Reference Application

When the model is uploaded the next step is to replace the default sensors and the data.

Placing sprites

The sprites are two dimensional in the model and represent the sensors. In the first place the device's models need to be defined in the device-models.json. Each device model contains: an Id, name, description, type, unit, minimum range value and maximum range value. The device inserted are the one listed in Table 1.

```
[
  {
    "deviceModelId": "d370a293-4bd5-4bdb-a3df-376dc131d44c",
    "deviceModelName": "Human Comfort Sensor",
    "deviceModelDesc": "Monitors indoor air quality by measuring levels of
Carbon Dioxide (CO2), temperature, and humidity.",
    "deviceProperties": [
      {
        "propertyId": "Temperature",
        "propertyName": "Temperature",
        "propertyDesc": "Temperature in Celsius",
        "propertyType": "double",
        "propertyUnit": "Celsius",
        "rangeMin": "-40.00",
        "rangeMax": "80.00"
      },
      {
        "propertyId": "Humidity",
        "propertyName": "Humidity",
        "propertyDesc": "Relative humidity in percentage",
        "propertyType": "double",
        "propertyUnit": "%RH",
        "rangeMin": "0.00",
        "rangeMax": "100.00"
      },
      {
        "propertyId": "Luminosity",
        "propertyName": "Luminosity",
        "propertyDesc": "Luminosity",
        "propertyType": "double",
        "propertyUnit": "Lux",
        "rangeMin": "300.00",
        "rangeMax": "400.00"
      },
      {
        "propertyId": "Temp_MPU",
        "propertyName": "Temp_MPU",
        "propertyDesc": "Temperature of Fancoil's motor",
        "propertyType": "double",
        "propertyUnit": "Celsius",
        "rangeMin": "-40.00",
        "rangeMax": "85.00"
      },
      {
        "propertyId": "A-X",
        "propertyName": "A-X",
        "propertyDesc": "Fancoil's motor vibration",
        "propertyType": "double",
        "propertyUnit": "°/s",

```

```
        "rangeMin": "0",
        "rangeMax": "2000.00"
    },
    {
        "propertyId": "A-Y",
        "propertyName": "A-Y",
        "propertyDesc": "Fancoil's motor vibration",
        "propertyType": "double",
        "propertyUnit": "°/s",
        "rangeMin": "0",
        "rangeMax": "2000.00"
    },
    {
        "propertyId": "A-Z",
        "propertyName": "A-Z",
        "propertyDesc": "Fancoil's motor vibration",
        "propertyType": "double",
        "propertyUnit": "°/s",
        "rangeMin": "0",
        "rangeMax": "2000.00"
    },
    {
        "propertyId": "G-X",
        "propertyName": "G-X",
        "propertyDesc": "Fancoil's motor vibration",
        "propertyType": "double",
        "propertyUnit": "g",
        "rangeMin": "2.00",
        "rangeMax": "16.00"
    },
    {
        "propertyId": "G-Y",
        "propertyName": "G-Y",
        "propertyDesc": "Fancoil's motor vibration",
        "propertyType": "double",
        "propertyUnit": "g",
        "rangeMin": "2.00",
        "rangeMax": "16.00"
    },
    {
        "propertyId": "G-Z",
        "propertyName": "G-Z",
        "propertyDesc": "Fancoil's motor vibration",
        "propertyType": "double",
        "propertyUnit": "g",
        "rangeMin": "2.00",
        "rangeMax": "16.00"
    },
    {
```

```
    "propertyId": "Voltage_1",
    "propertyName": "Voltage_1",
    "propertyDesc": "fancoil's motor voltage",
    "propertyType": "double",
    "propertyUnit": "Volt",
    "rangeMin": "0.00",
    "rangeMax": "230.00"
  },
  {
    "propertyId": "Voltage_2",
    "propertyName": "Voltage_2",
    "propertyDesc": "fancoil's motor voltage",
    "propertyType": "double",
    "propertyUnit": "Volt",
    "rangeMin": "0.00",
    "rangeMax": "230.00"
  },
  {
    "propertyId": "Voltage_3",
    "propertyName": "Voltage_3",
    "propertyDesc": "fancoil's motor voltage",
    "propertyType": "double",
    "propertyUnit": "Volt",
    "rangeMin": "0.00",
    "rangeMax": "230.00"
  },
  {
    "propertyId": "T1",
    "propertyName": "T1",
    "propertyDesc": "Delivery water temperature",
    "propertyType": "double",
    "propertyUnit": "Celsius",
    "rangeMin": "0.00",
    "rangeMax": "90.00"
  },
  {
    "propertyId": "T2",
    "propertyName": "T2",
    "propertyDesc": "Return water temperature",
    "propertyType": "double",
    "propertyUnit": "Celsius",
    "rangeMin": "0.00",
    "rangeMax": "90.00"
  },
  {
    "propertyId": "T3",
    "propertyName": "T3",
    "propertyDesc": "Inlet air temperature",
    "propertyType": "double",
```

```
    "propertyUnit": "Celsius",
    "rangeMin": "0.00",
    "rangeMax": "90.00"
  },
  {
    "propertyId": "T4",
    "propertyName": "T4",
    "propertyDesc": "Motor case temperature",
    "propertyType": "double",
    "propertyUnit": "Celsius",
    "rangeMin": "0.00",
    "rangeMax": "90.00"
  },
  {
    "propertyId": "RTD",
    "propertyName": "RTD",
    "propertyDesc": "Higher temperature between T1,T2,T3,T4",
    "propertyType": "double",
    "propertyUnit": "Celsius",
    "rangeMin": "-200.00",
    "rangeMax": "500.00"
  },
  {
    "propertyId": "Irms1",
    "propertyName": "Irms1",
    "propertyDesc": "Current",
    "propertyType": "double",
    "propertyUnit": "mA",
    "rangeMin": "0.00",
    "rangeMax": "100.00"
  },
  {
    "propertyId": "Irms2",
    "propertyName": "Irms2",
    "propertyDesc": "Current",
    "propertyType": "double",
    "propertyUnit": "mA",
    "rangeMin": "0.00",
    "rangeMax": "100.00"
  },
  {
    "propertyId": "Irms3",
    "propertyName": "Irms3",
    "propertyDesc": "Current",
    "propertyType": "double",
    "propertyUnit": "mA",
    "rangeMin": "0.00",
    "rangeMax": "100.00"
  },
}
```

```
{
  "propertyId": "RP1",
  "propertyName": "RP1",
  "propertyDesc": "Real power",
  "propertyType": "double",
  "propertyUnit": "Watt",
  "rangeMin": "-2.00",
  "rangeMax": "90.00"
},
{
  "propertyId": "RP2",
  "propertyName": "RP2",
  "propertyDesc": "Real power",
  "propertyType": "double",
  "propertyUnit": "Watt",
  "rangeMin": "-2.00",
  "rangeMax": "90.00"
},
{
  "propertyId": "RP3",
  "propertyName": "RP3",
  "propertyDesc": "Real power",
  "propertyType": "double",
  "propertyUnit": "Watt",
  "rangeMin": "-2.00",
  "rangeMax": "90.00"
},
{
  "propertyId": "Voltage_room",
  "propertyName": "Voltage_room",
  "propertyDesc": "Real power",
  "propertyType": "double",
  "propertyUnit": "Volt",
  "rangeMin": "0.00",
  "rangeMax": "1000.00"
},
{
  "propertyId": "Current_room",
  "propertyName": "Current_room",
  "propertyDesc": "Current",
  "propertyType": "double",
  "propertyUnit": "mA",
  "rangeMin": "0.00",
  "rangeMax": "100.00"
},
{
  "propertyId": "Power_room",
  "propertyName": "Power_room",
  "propertyDesc": "Power",
```



```
        "propertyType": "double",  
        "propertyUnit": "Watt",  
        "rangeMin": "0.00",  
        "rangeMax": "90.00"  
      }  
    ]  
  }  
]
```

Figure 48 Code of device model properties

Once the device-model.json is set the placing of the devices needs to be defined. The coordinates of the devices are defined in devices.json and each device has an id and a name. The position is defined by the x, y, z coordinates. In order to have the correct position of the sprites the playground tool was used. It's accessible through `localhost:9000/playground`. This tool displays our model and makes possible the positioning of temporary sprites. The coordinates of the sprites in JSON format can be visualized and copied in our application code.

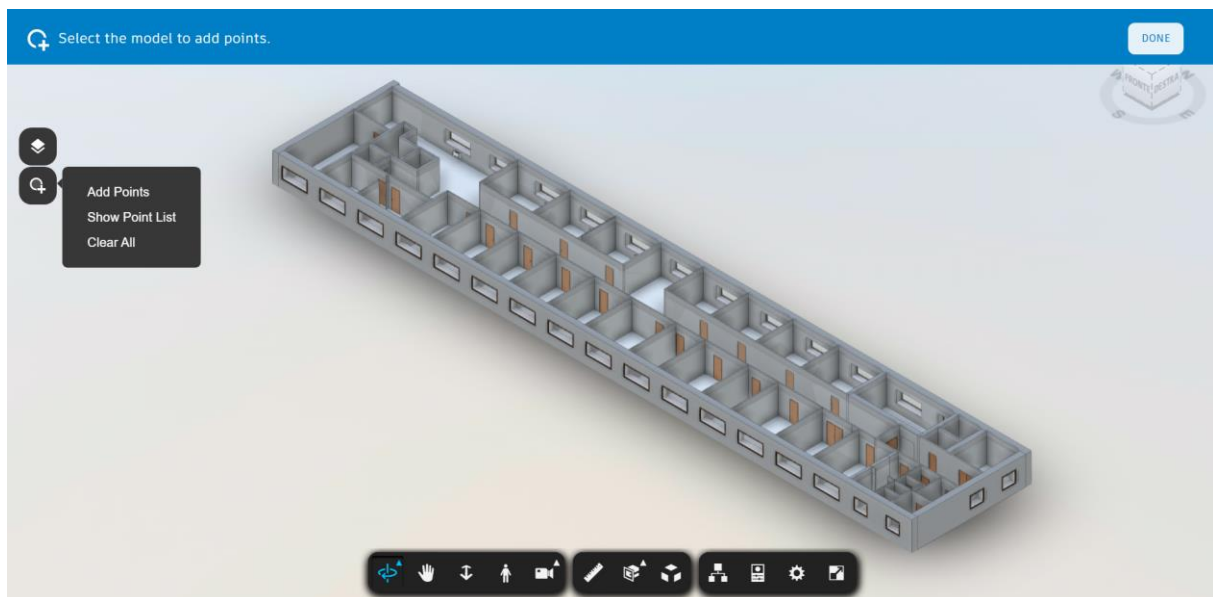


Figure 49 localhost:9000/playground with our model

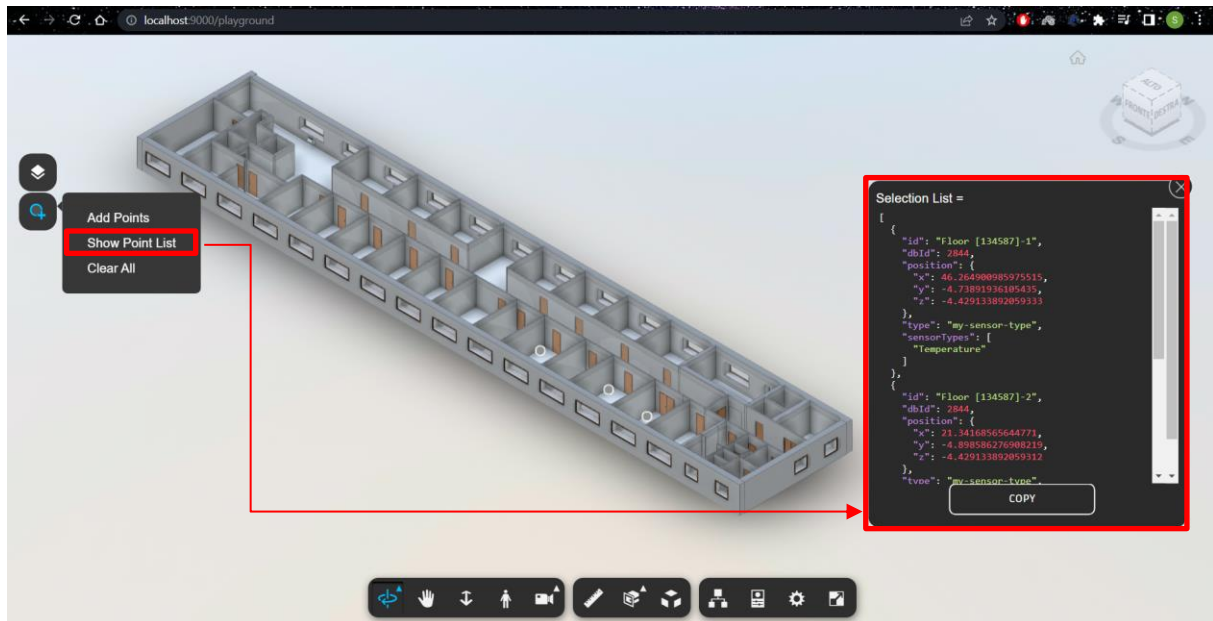


Figure 50 localhost:9000/playground with sensors' coordinates in JSON format

The JSON obtained can be then copied and modified in devices.json file.



Figure 51 Devices.json with coordinates of a sprite

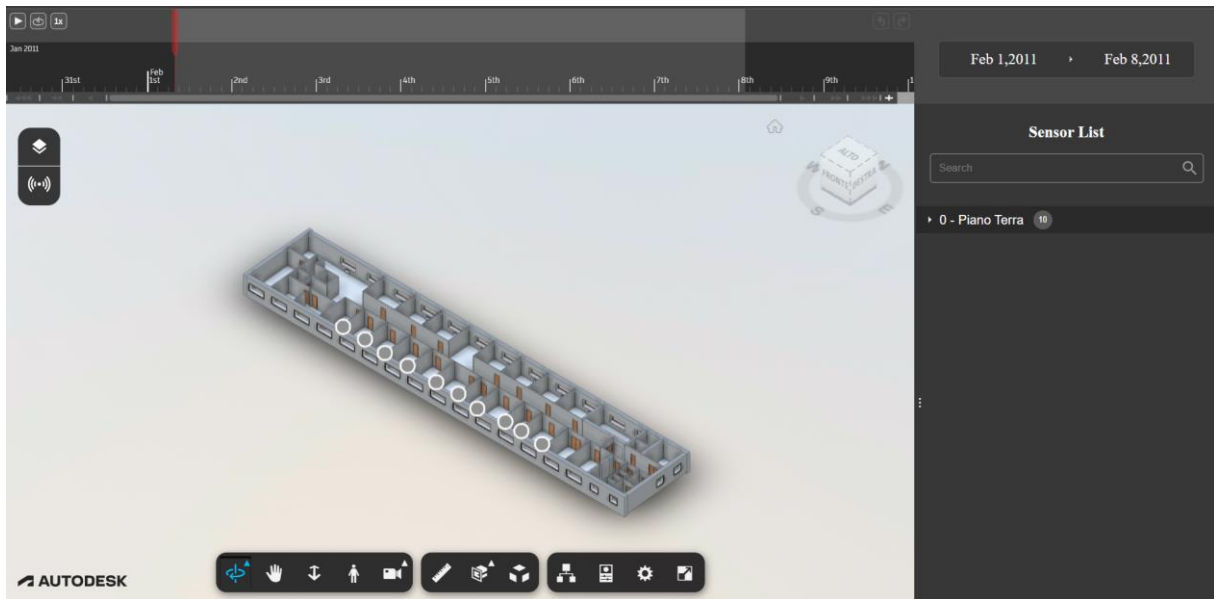


Figure 52 Placed sprites

Importing CSV data into the model

In this case the time-series data source to the Data Visualization was a CSV file. This file was inserted in the csv folder at the path: `forge-dataviz-iot-reference-app\server\gateways\csv`. The CSV file was set so that every measure was separated by a semicolon “;”. The time format was changed since the one required from the Forge app must have the following format:

“YYYY-MM-DDTHH:MM:SS.000Z”

At “`forge-dataviz-iot-reference-app\server\router\DataAPI.js`” at the code line 30 the variable “`CSV.DELIMITER`” was changed from “`/t`” to “`;`” since, as previously stated, the value delimiter in the CSV file was a semicolon.

```
30- const delimiter = process.env.CSV_DELIMITER || "\t"; 30+ const delimiter = process.env.CSV_DELIMITER || ";";
```

Figure 53 Line 30 of the code at “`forge-dataviz-iot-reference-app\server\router\DataAPI.js`”

The name of the CSV file must be the same of the variable “`Id`” in the `devices.json` file, this enables the association of the CSV file to a particular sprite. In this case the name of the CSV file was “`Hyperion_24`”, as the “`Id`” of the placed sprite.

One last step is the set of the environmental variables to display the data in the Forge viewer. In this case as well, they were set from the Windows system and are the following:

- `ADAPTER_TYPE` must be set to “`csv`”;

- CSV_MODEL_JSON must be set to the device-models.json file's location, in particular the relative path has to be indicated;
- CSV_DEVICE_JSON must be set to the devices.json file's location, in particular the relative path has to be indicated;
- CSV_FOLDER must be set to the CSV folder/directory location, in particular the relative path has to be indicated;
- CSV_DATA_START which indicates the start data of the CSV file and must be in the format YYYY-MM-DDTHH:MM:SS.000Z;
- CSV_DATA_END which indicates the end data of the CSV file and must be in the format YYYY-MM-DDTHH:MM:SS.000Z;

In the following figure are indicate the environmental variable needed for the uploading of the data.

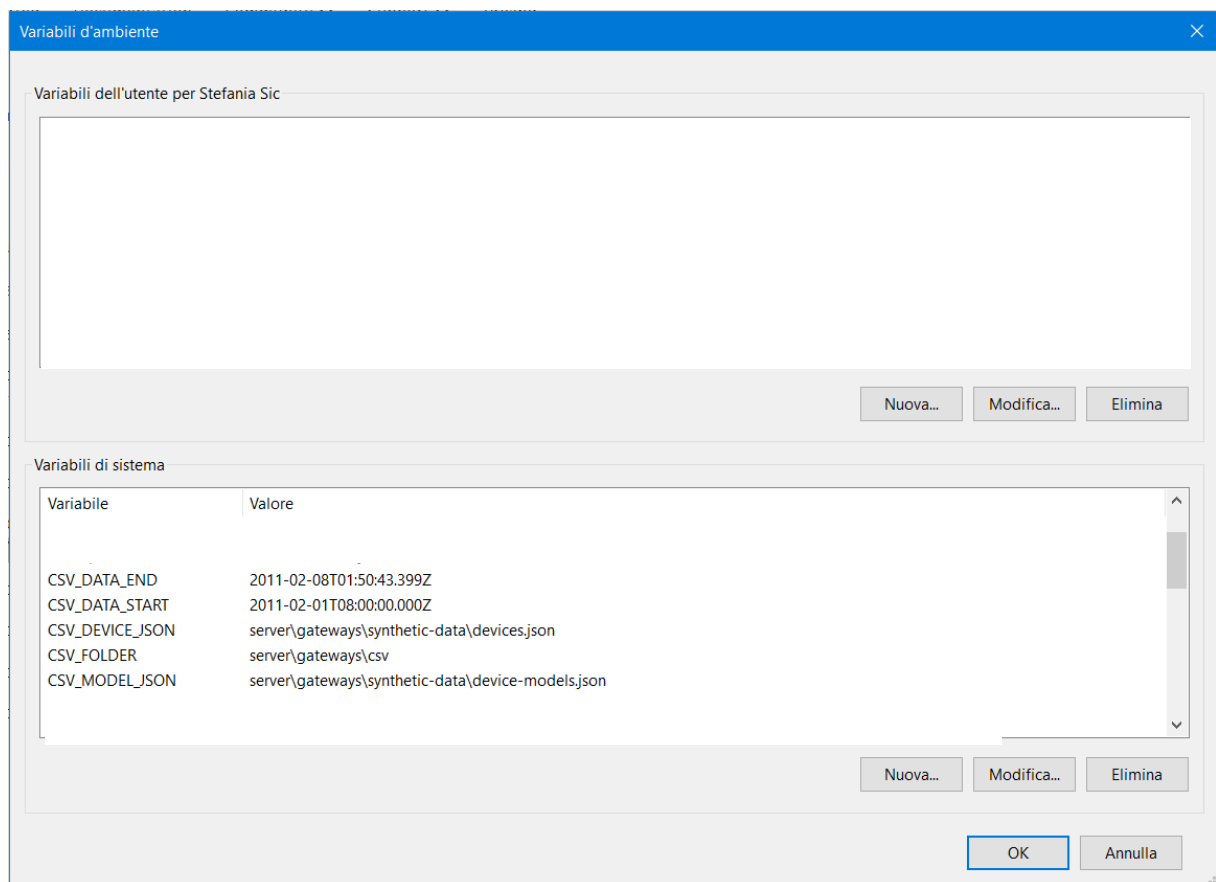


Figure 54 Environmental variables for the CSV file

Adding sensors' icon

After importing the data, it was necessary the insertion of icon to be associated to every parameter in order to be displayed in the Forge Viewer.

In “Dot.jsx” were added new “SensorStyleDefinitions”, defining URL, with relative path, and colour.

```
forge-dataviz-iot-reference-app > client > pages > Dot.jsx > Dot > SensorStyleDefinitions > voltage > url
45  /**
46   * @type {SensorStyleDefinitions}
47   */
48  const SensorStyleDefinitions = {
49    co2: {
50      url: `${ApplicationContext.assetUrlPrefix}/images/co2.svg`,
51      color: 0xffffffff,
52    },
53    temperature: {
54      url: `${ApplicationContext.assetUrlPrefix}/images/thermometer.svg`,
55      color: 0xffffffff,
56    },
57    fancoil: {
58      url: `${ApplicationContext.assetUrlPrefix}/images/fan-00.svg`,
59      color: 0xffffffff,
60    },
61    luminoosity: {
62      url: `${ApplicationContext.assetUrlPrefix}/images/brightness-low-fill-svgrepo-com.svg`,
63      color: 0xffffffff,
64    },
65    fancoil2: {
66      url: `${ApplicationContext.assetUrlPrefix}/images/fan-02.svg`,
67      color: 0xffffffff,
68    },
69    default: {
70      url: `${ApplicationContext.assetUrlPrefix}/images/circle.svg`,
71      color: 0xffffffff,
72    },
73    current: {
74      url: `${ApplicationContext.assetUrlPrefix}/images/energy.svg`,
75      color: 0xffffffff,
76    },
77    voltage: {
78      url: `${ApplicationContext.assetUrlPrefix}/images/voltage.svg`,
79      color: 0xffffffff,
80    },
81  };
```

Figure 55 Code snippet for defining sensors' Propertyiconmap in Dot.jsx

In “SenorStyles.js” file, the image can be imported and associated to the relative parameter. The icon format employed was .svg.

In order to be displayed correctly small size .svg file were imported, 24x24 pixel.

```

forge-dataviz-iot-reference-app > client > config > JS SensorStyles.js > PropertyIconMap
1  import circleSvg from "../../assets/images/circle.svg";
2  import circleHighlightedSvg from "../../assets/images/circle_highlighted.svg";
3  import temperatureSvg from "../../assets/images/temperature_property.svg";
4  import humiditySvg from "../../assets/images/humidity_property.svg";
5  import co2Svg from "../../assets/images/co2_property.svg";
6  import fancoil from "../../assets/images/fan-00.svg";
7  import luminosity from "../../assets/images/brightness-low-fill-svgrepo-com.svg";
8  import fancoil2 from "../../assets/images/fan-02.svg";
9  import current from "../../assets/images/energy.svg";
10 import voltage from "../../assets/images/voltage.svg";
11

```

Figure 56 Code snippet in `sensorstyle.js` for defining sensors' image of `PropertyIconmap`

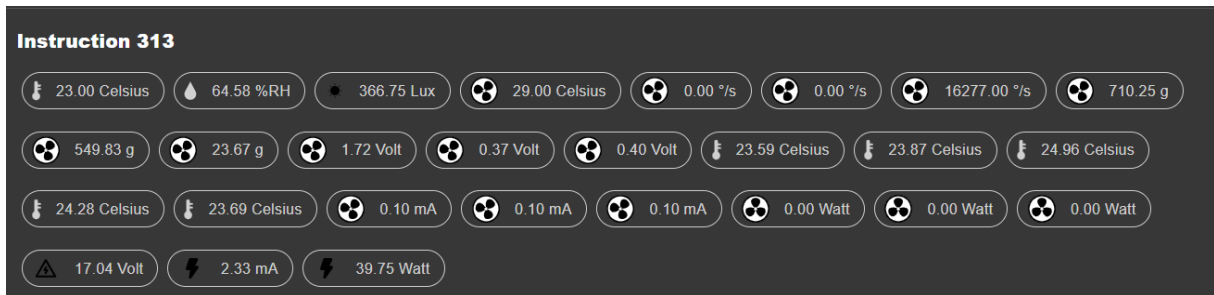


Figure 57 Icon displayed on Forge in the “sensor list”

```

forge-dataviz-iot-reference-app > client > config > JS SensorStyles.js > PropertyIconMap
37  'CO2': [0x1e9600, 0x111200, 0x110000],
38  };
39
40  export const PropertyIconMap = {
41    Temperature: temperatureSvg,
42    Humidity: humiditySvg,
43    "CO2": co2Svg,
44    "Temp_MPU": temperatureSvg,
45    "T1": temperatureSvg,
46    "T2": temperatureSvg,
47    "T3": temperatureSvg,
48    "T4": temperatureSvg,
49    "RTD": temperatureSvg,
50    "RP1": fancoil,
51    "RP2": fancoil,
52    "RP3": fancoil,
53    "Luminosity": luminosity,
54    "Temp_MPU": fancoil2,
55    "Power_room": current,
56    "A-X": fancoil2,
57    "A-Y": fancoil2,
58    "A-Z": fancoil2,
59    "G-X": fancoil2,
60    "G-Y": fancoil2,
61    "G-Z": fancoil2,
62    "Voltage_1": fancoil2,
63    "Voltage_2": fancoil2,
64    "Voltage_3": fancoil2,
65    "Irms1": fancoil2,
66    "Irms2": fancoil2,
67    "Irms3": fancoil2,
68    "Current_room": current,
69    "Voltage_room": voltage,
70  };
71

```

Figure 58 Code snippet associating icon to each parameter in “`SensorStyles.js`”

It is also possible to define the colour of the gradient for each parameter in “SensorStyles.js”

```

34 export const PropIdGradientMap = {
35   Temperature: [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
36   Humidity: [0x00f260, 0x0575e6],
37   "CO2": [0x1e9600, 0xffff200, 0xff0000],
38   "RTD": [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
39   "Temp_MPU": [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
40   "T1": [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
41   "T2": [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
42   "T3": [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
43   "T4": [0x0000ff, 0x00ff00, 0xffff00, 0xff0000],
44 };
45

```

Figure 59 Code snippet for defining colours of parameters' gradient

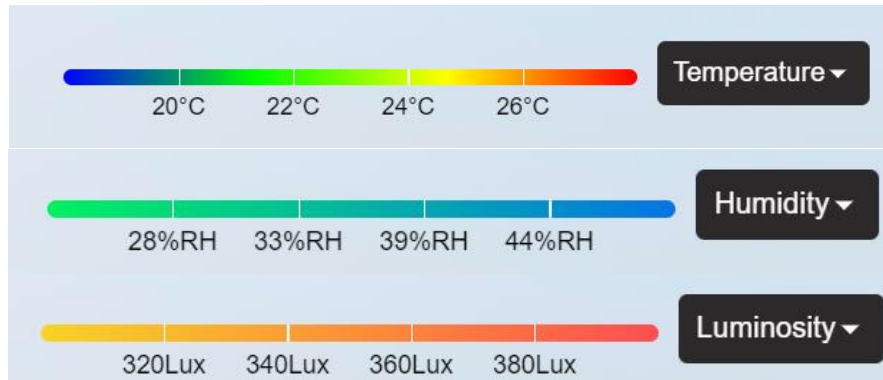


Figure 60 Parameters' gradient for heatmap

5.3 Data visualization

Once the 3D model is uploaded, the sprites are positioned and the data are uploaded in the Forge app, the application can be launched through the command “npm run dev”.

When the application is launched the static address <http://localhost:9000/> can be opened.

The sensor list contains all the sprites that are ordered according to the levels and the rooms in which sprites are placed.

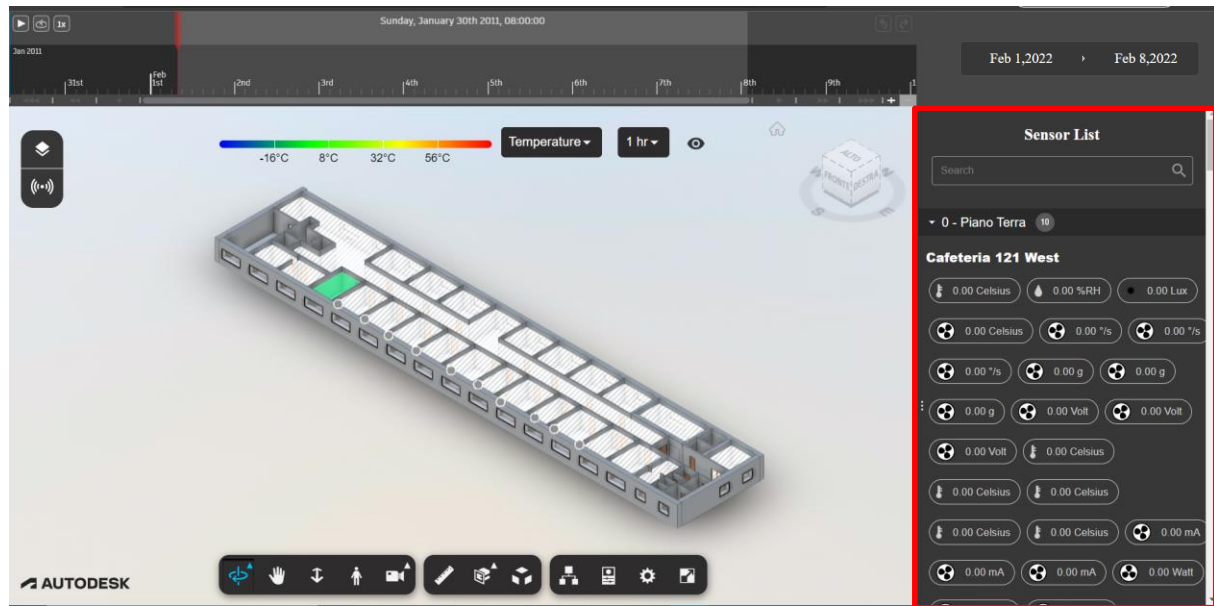
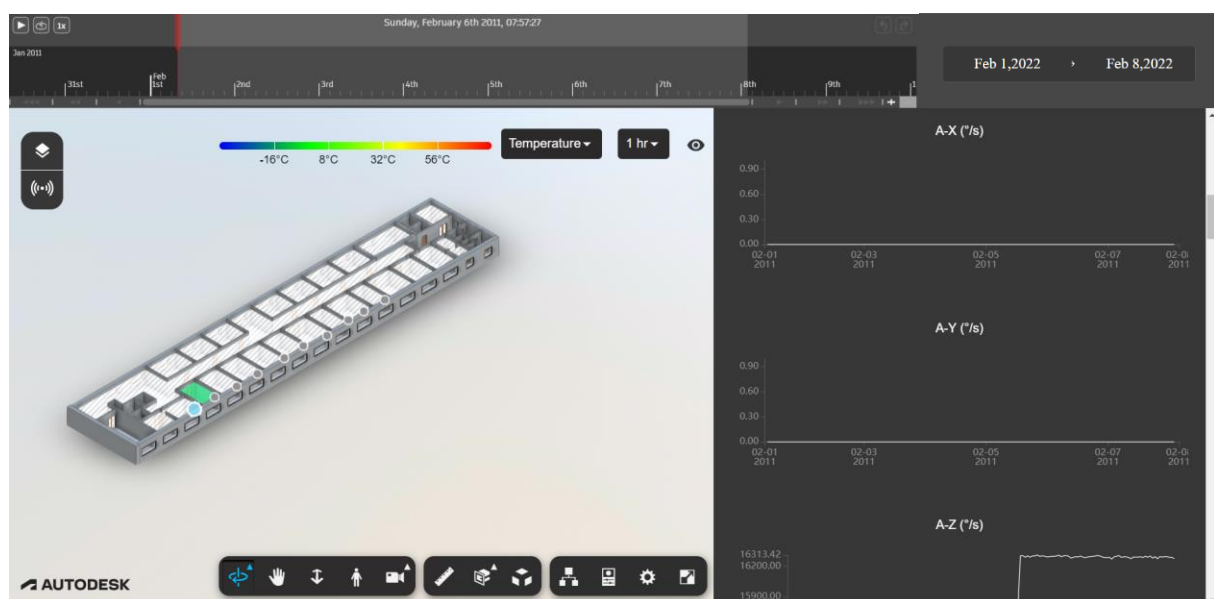
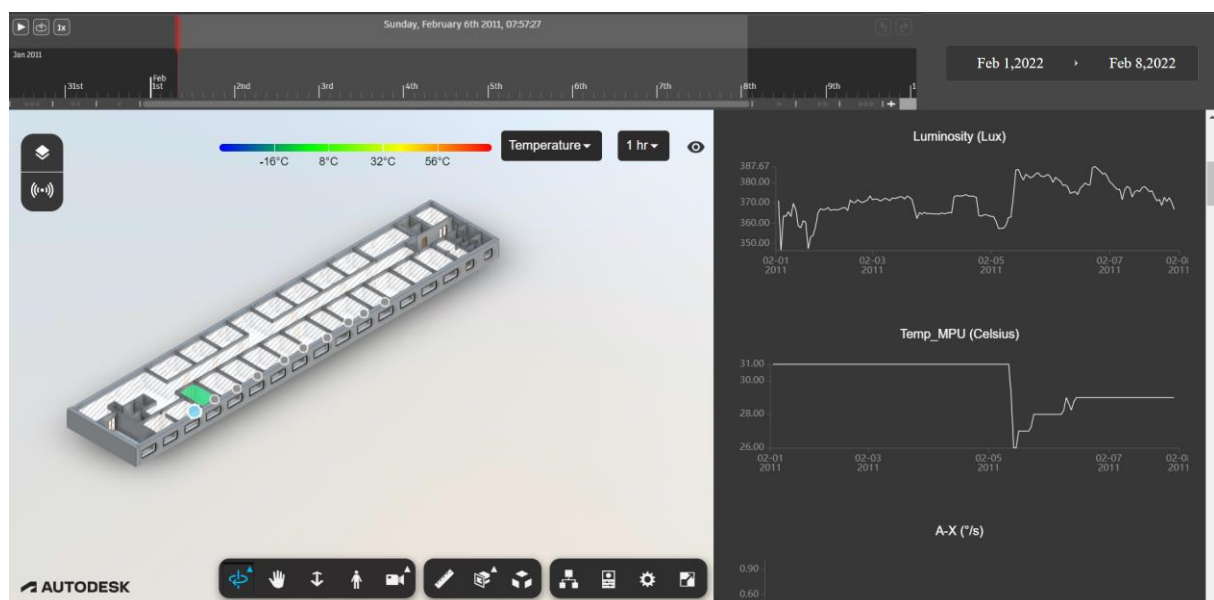
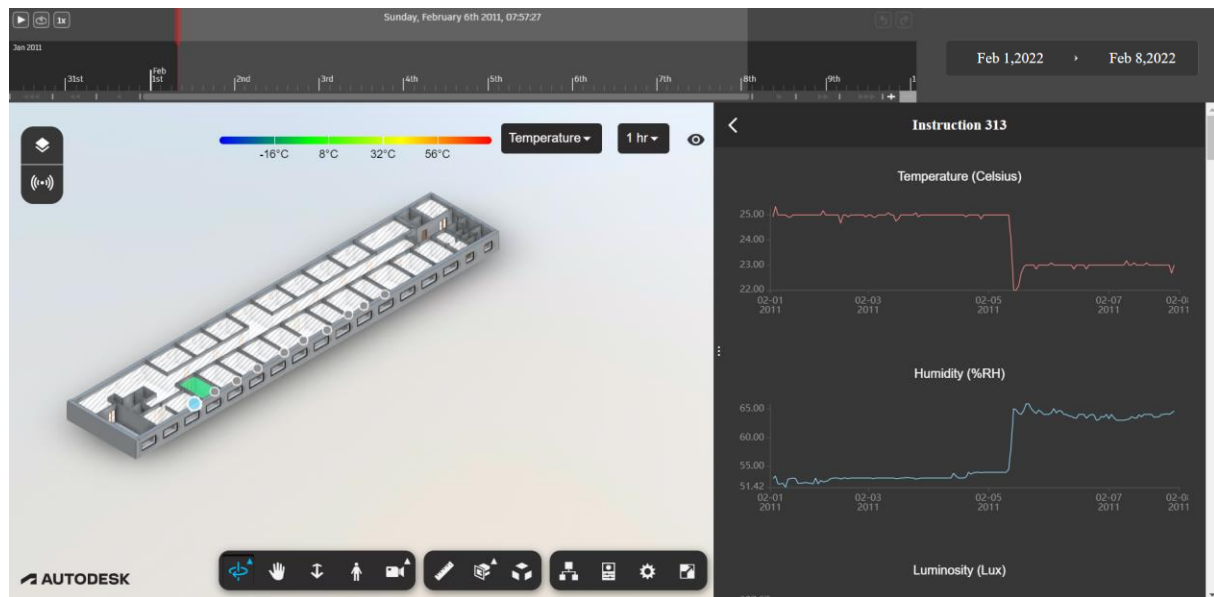
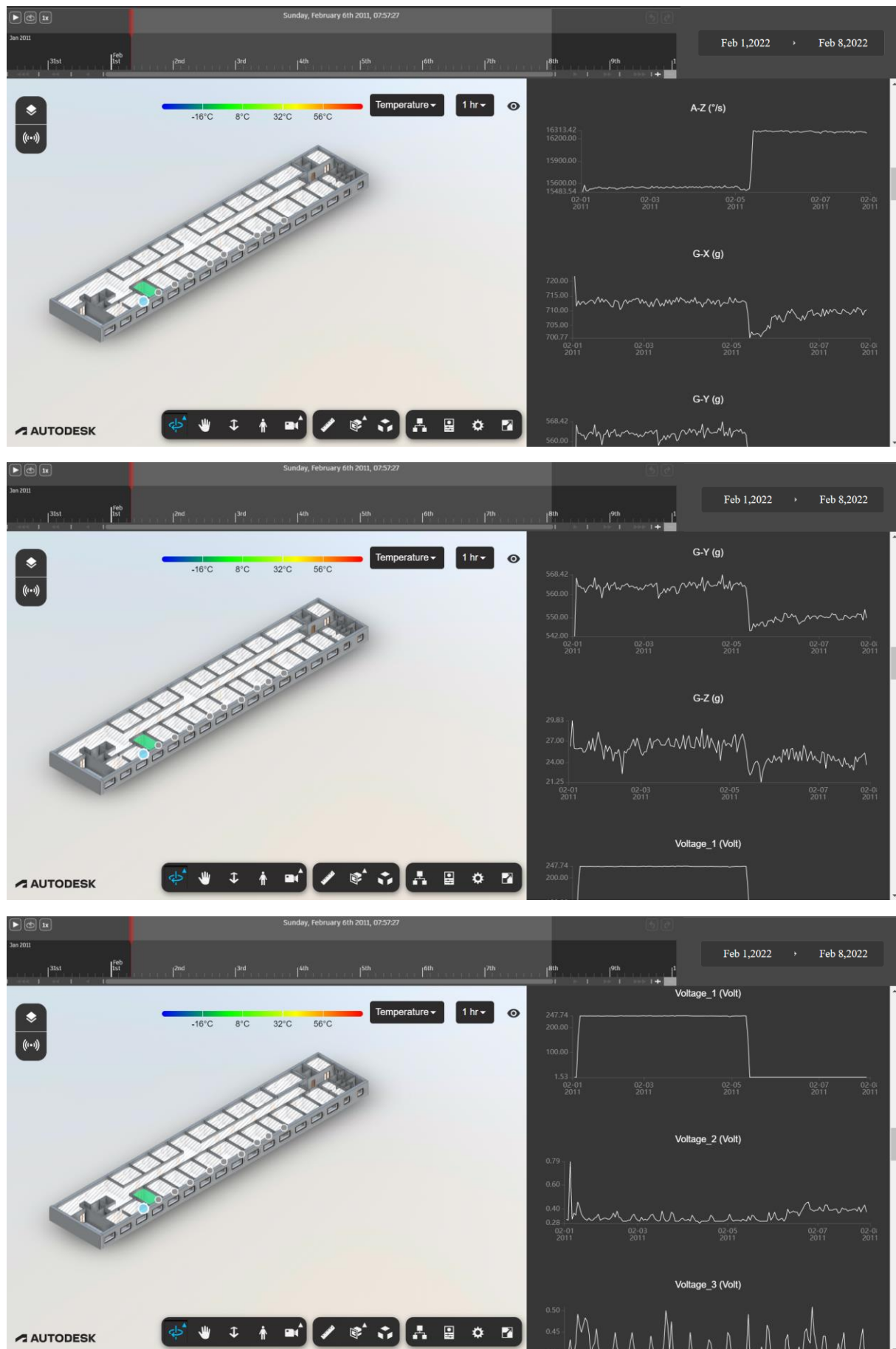
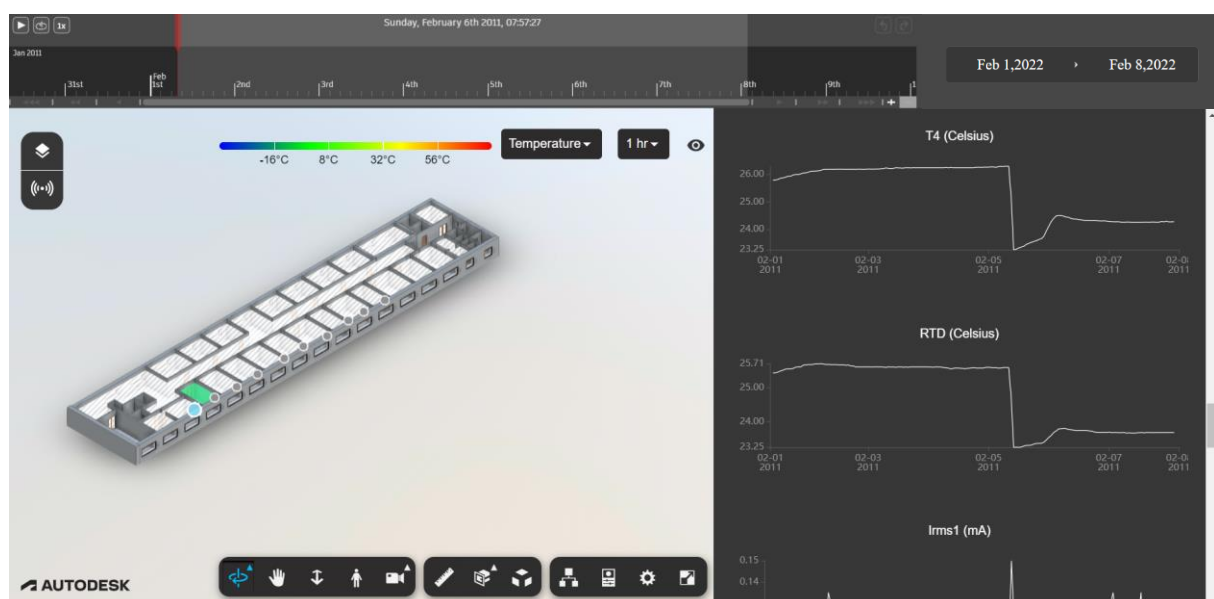
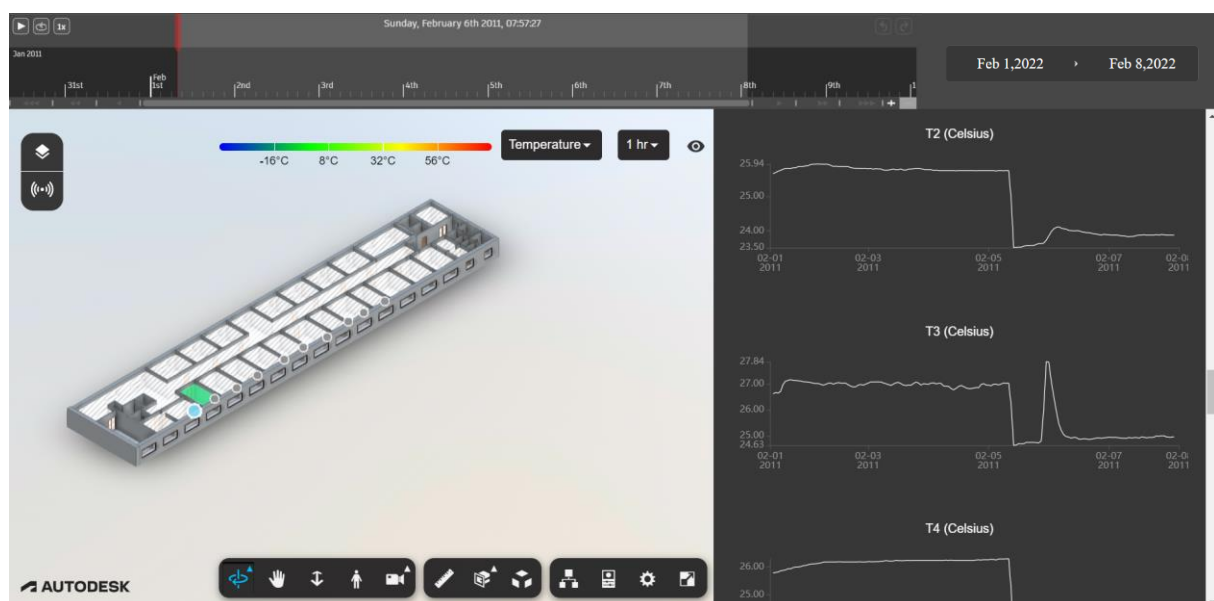
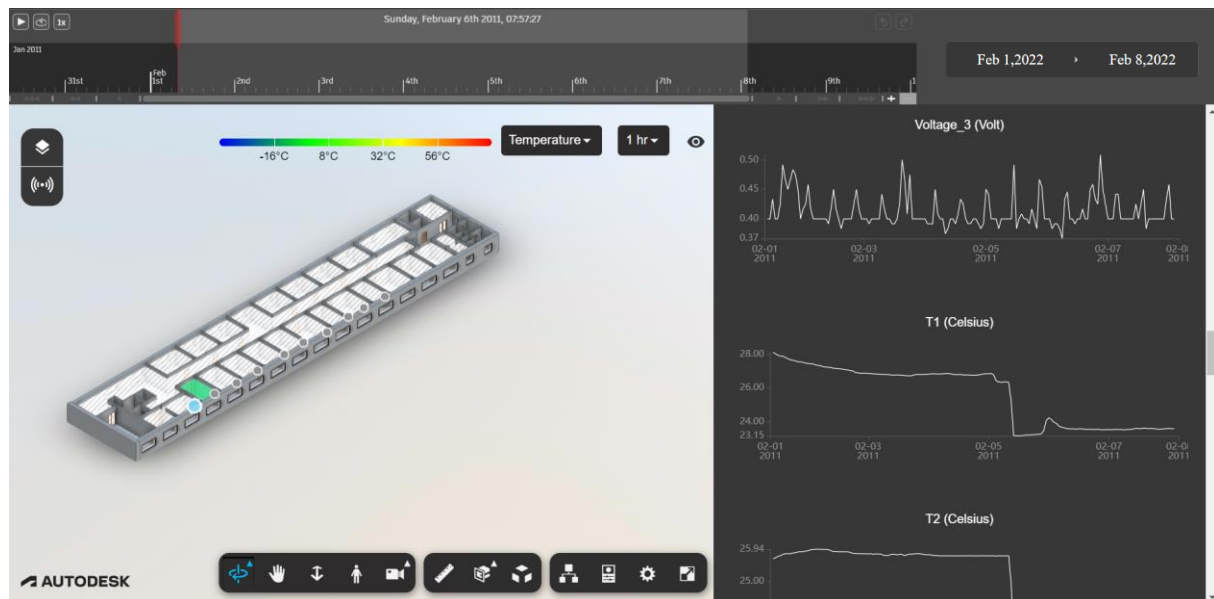


Figure 61 Sensor list with sprites divided by level and room in which are placed

In the following figures are displayed the model with the relative sensors' data through time. Graphs that describes data variation through time are available and the range of time in which parameters' variation is displayed can be selected.







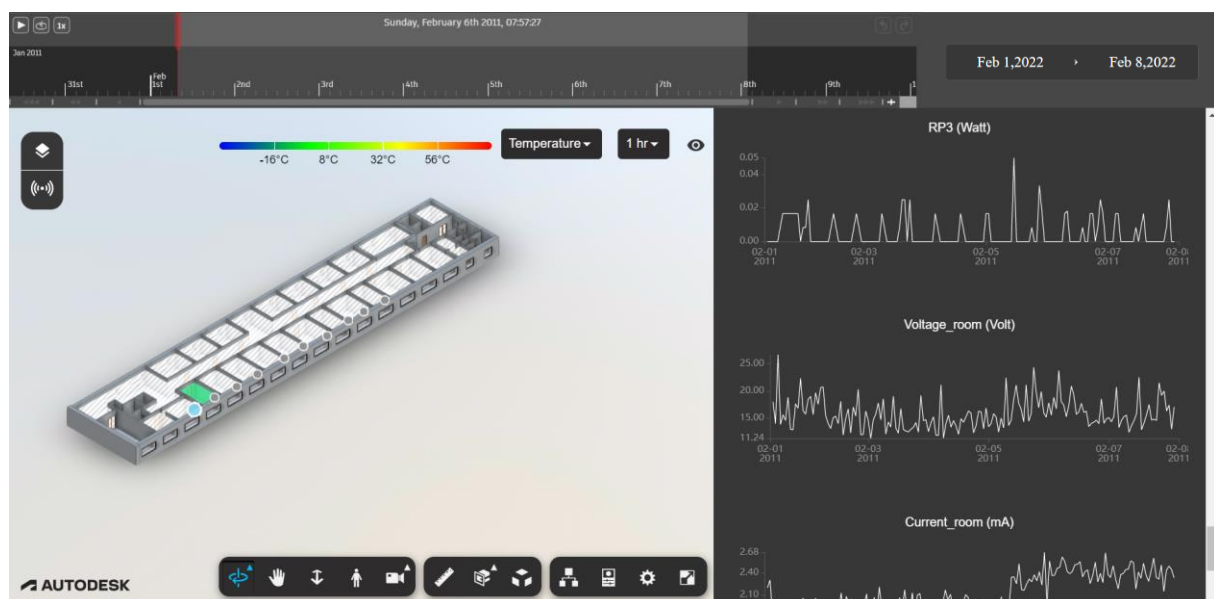
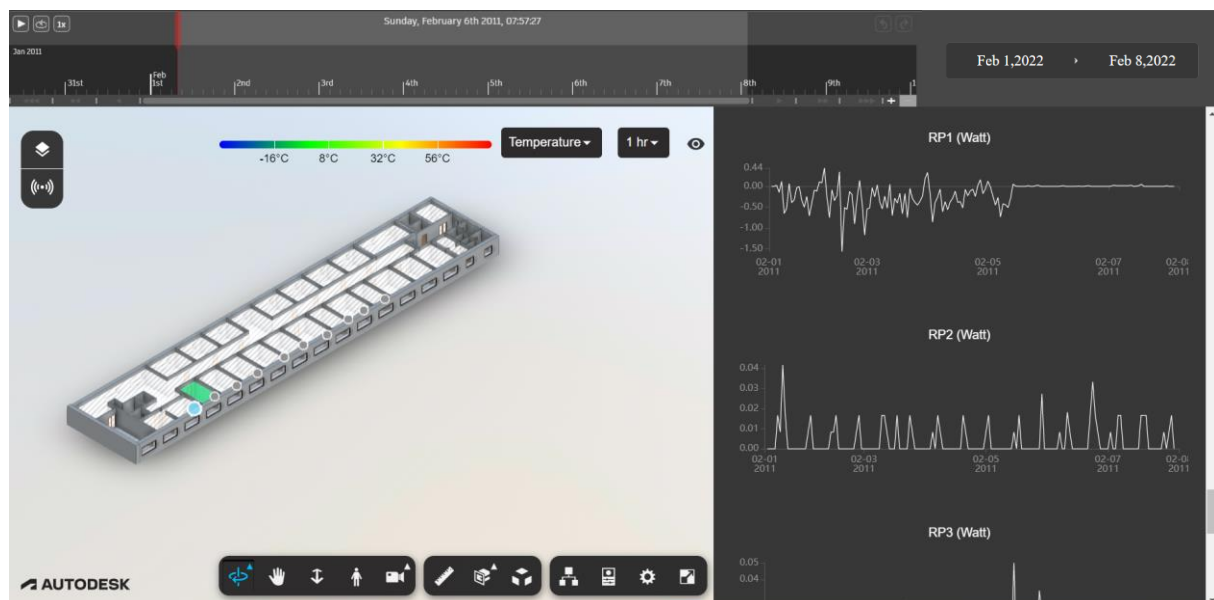
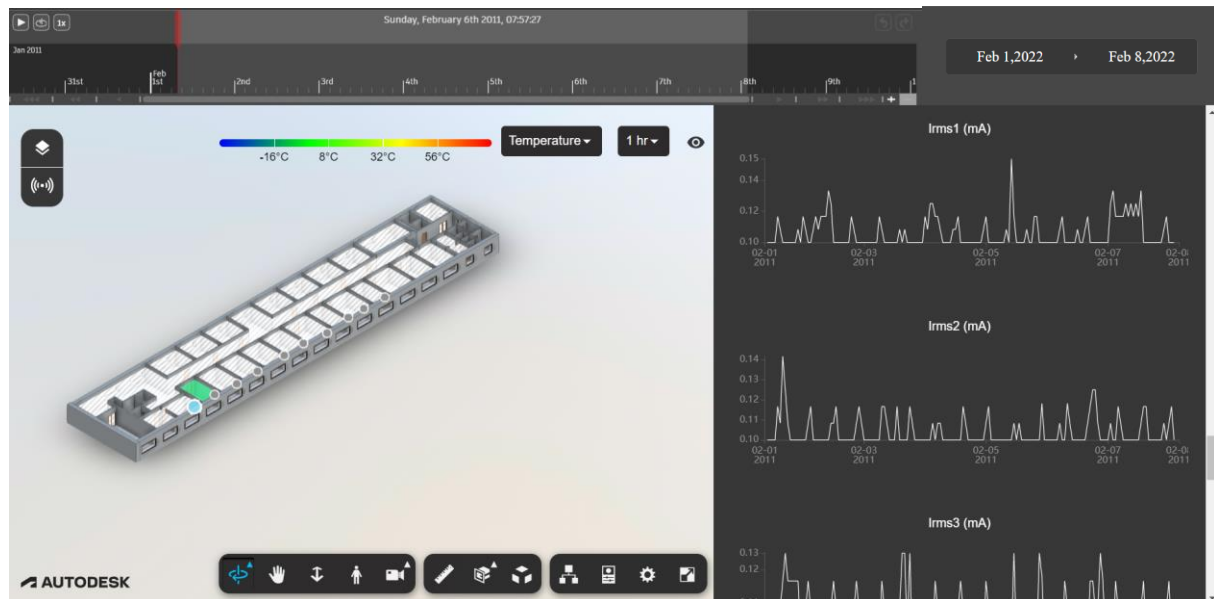




Figure 62 Graphs' parameter-time

When passing over the sprite with the cursor a preview of the data can be displayed.

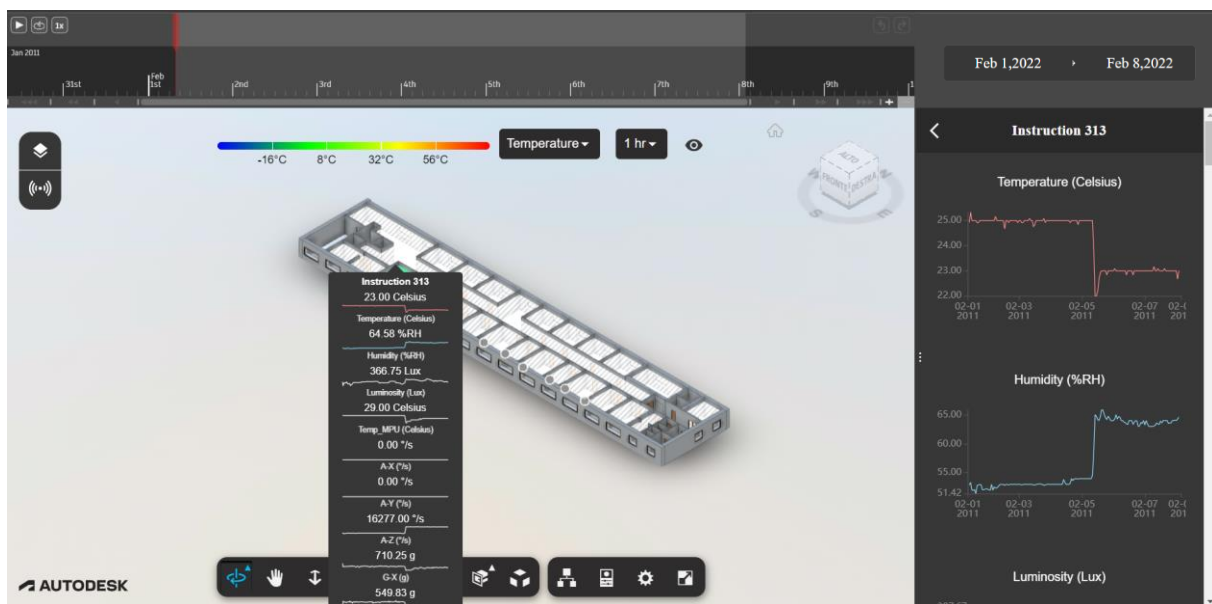


Figure 63 Preview of data

Also, when passing with the cursor over the graphs it can be seen in detail the values at a specific time.

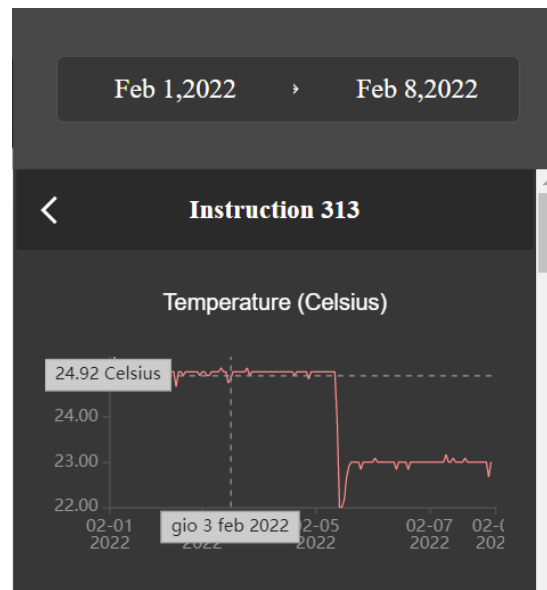


Figure 64 Value at a specific time

The application enables, also, the viewing of heatmap connected to data, which gives for example, a quick insight of the internal room temperature condition.

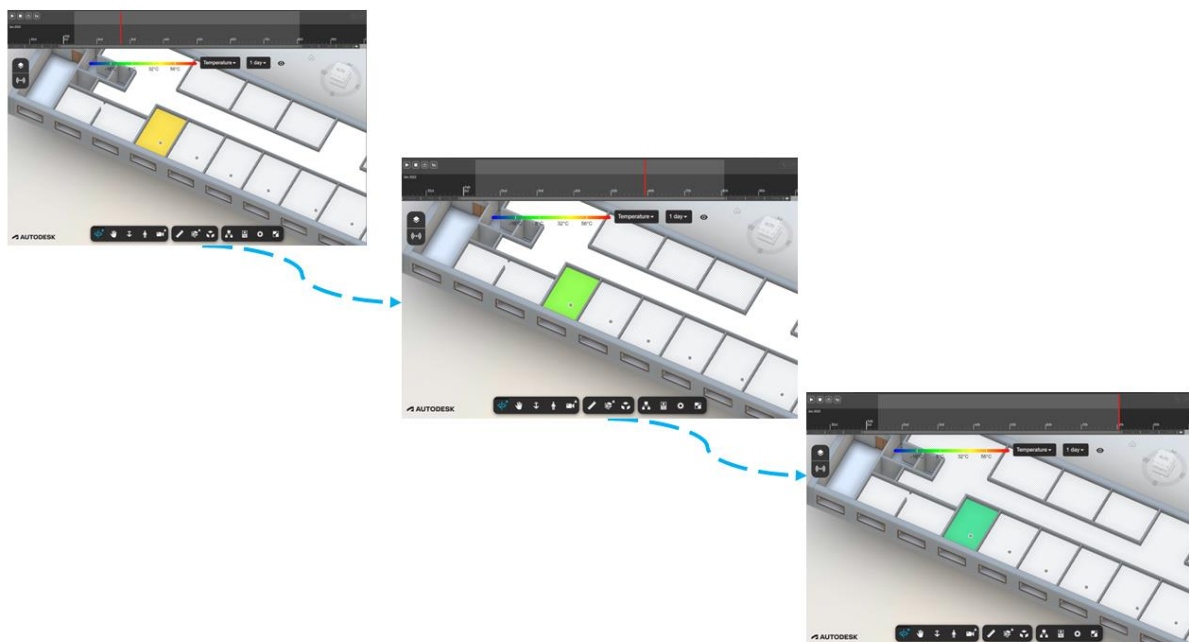


Figure 65 Heatmap variation for temperature on Autodesk Forge

6 Conclusion

This study showed how the use of cloud-based platform for FM has a lot of potential for practical use and can be a tool to achieve data visualization within BIM model.

In the first part of this study were discussed the concepts at the base of the topic of this thesis. Successively a study was conducted on the state of art of BIM-IoT integration, thanks to the consultation of studies and research. In this section emerged that a lot of studies regarding BIM-IoT integration were conducted, but few tested the potential of cloud-based platforms. Therefore cloud-based platforms, that enable BIM-IoT integration, were introduced in order to select one to be employed in the case study.

Eventually a BIM-IoT architecture was proposed, which would allow the practical integration between the sensors' data and BIM model, through the use of a cloud-based platform, Autodesk Forge. Its application was further analyzed in the case study section, which permitted the visualization of real-time data for buildings' asset monitoring. The data visualization given by Autodesk Forge permits to know the real time condition of the building within the BIM model. In this context BIM represent the spatial and static information, whereas sensors' data represents the dynamic changes of condition of buildings' asset. This type of visualization can be a support for FM activities and decision making, since the data monitoring gives material to justify decisions' process and bring a change of maintenance practice towards predictive maintenance.

The BIM-IoT data visualization's benefits for FM activities can be summarized in:

- Detect anomalies and pursue maintenance action before failure occurs, avoiding interruption of service in the building;
- Spatially locate the failure for a precise and functional maintenance operation;
- Analysis of sensors' data for planning and schedule of maintenance operations in advance.

While the benefits of the proposed framework are many, some limitations must be highlighted. The application used, Autodesk Forge, to be used in your own project must be

customized, which requires programming skills. It is, also, an application in a development state and during its set up many problems can occur. Besides, some parts of the “Data visualization’s developers guide”, necessary for running the application, on Forge site need to be reviewed or updated, since the following of some steps cannot guarantee the success of the operations during the application’s set up. These problems can cause the necessity of a long time for the setup of the application. In addition, some steps in the guide are taken for granted, even if the final user of the application cannot be familiar with coding.

Another limitation was the number of sensors employed for the framework’s application, which was limited and restricted to a single room. Larger range sensors and in a larger number should be tested in the proposed system.

This thesis can be the starting point for future researches. It can be studied the possibility to integrate the Forge application with other cloud platform, like AWS and Microsoft Azure. In fact, it is possible to create database in Azure, for example, where the sensors’ data can be stored and taken to be displayed in the Forge application.

Also, the Forge application can be hosted in these cloud platforms, in order to make easier the access to the application from others device. In fact, it is possible to store the Forge’s directory in Azure’s cloud, for example, and then run the application, without the necessity to have app’s directory in the local disk of the device.

Even if there are some limitations to overcome, this study demonstrates the potential of using BIM-IoT integration systems for FM activities and improvements can be done to make this framework applicable at a practical level.

7 Bibliography and web references

- [1] M. Stefan, “ACE,” [Online]. Available: https://www.ace-cae.eu/fileadmin/New_Upload/3._Area_2_Practice/BIM/Other_Docs/1_S.Mordue_Definition_of_BIM_01.pdf. [Accessed 11 06 2022].
- [2] “Biblus,” ACCA, [Online]. Available: <https://biblus.accasoftware.com/en/modeling-model-and-management-the-three-ms-of-bim-and-the-right-bim-tools/>. [Accessed 11 06 2022].
- [3] E. Toledo. [Online]. Available: <https://amamdouhsaad.wixsite.com/ibi-bim/post/the-history-of-bim>. [Accessed 11 06 2022].
- [4] [Online]. Available: <https://8dbim.weebly.com/history-of-bim.html>. [Accessed 11 06 2022].
- [5] S. S. Sinoh and F. Othman, “Review of BIM literature and government initiatives to promote BIM in,” in *2nd International Conference on Materials Technology and Energy*, 2020.
- [6] C. Allen and W. Shakantu, “The BIM revolution: a literature review on,” in *Proceedings of the 11 International Conference th*, 2016.
- [7] D. B. Hammad, A. G. Rishi and M. B. Yahaya, “MITIGATING CONSTRUCTION PROJECT RISK USING BUILDING INFORMATION MODELLING (BIM),” in *Waber*, Abuja, 2012.
- [8] T. Pavel, “An Investigation into the Possibilities of BIM and GIS Cooperation and Utilization of GIS in the BIM Process,” *Geoinformatics FCE CTU*, vol. 14, no. 1, p. 65, 2015.
- [9] S. Madakam, R. Ramaswamy and S. Tripathi, “Internet of Things (IoT): A Literature Review,” *Journal of Computer and Communications*, vol. 3, no. 5, pp. 164-173, 2015.
- [10] A. S. Gillis, “<https://www.techtarget.com/>,” IoT Agenda, [Online]. Available: <https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT>.

-
- [11] K. Ashton, "That 'internet of things' thing," *RFID Journal*, vol. 22, no. 7, pp. 97-114, 2009.
 - [12] S. Kuyoro, F. Osisanwo and O. Akinsowon, "Internet of things (IoT): an overview.," in *3rd International Conference on Advances in Engineering Sciences & Applied Mathematics (ICAESAM'2015)*, London (UK), 2015.
 - [13] T. Harwood, "Postscapes," 12 11 2019. [Online]. Available: <https://www.postscapes.com/iot-history/>. [Accessed 12 05 2022].
 - [14] O. Masud and M. Said, "Towards internet of things: Survey and future vision," *International Journal of Computer Networks*, vol. 5, no. 1, pp. 1-17, 2013.
 - [15] R. Khan, S. U. Khan, R. Zaheer and S. Khan, "Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges," in *In 2012 10th International Conference on Frontiers of Information Technology (FIT) pp257-260*, 2012.
 - [16] M. U. Farooq, M. Waseem, S. Mazhar, A. Khairi and T. Kamal, "A review on internet of things (IoT)," *International journal of computer applications*, vol. 113, no. 1, pp. 1-7, 2015.
 - [17] B. Li and J. Yu, "Research and Application on the Smart Home Based on Component Technologies and Internet of Things," *Procedia Engineering*, vol. 15, pp. 2087-2092, 2011.
 - [18] C. Sun, "Application of RFID Technology for Logistics on Internet of Things," *AASRI Procedia*, vol. 1, pp. 106-111, 2012.
 - [19] H. D. Kotha and V. M. Gupta, "IoT Application, A Survey," *International Journal of Engineering & Technology*, vol. 7, no. 2.7, pp. 891-896, 2018.
 - [20] L. Atzori, A. Iera and G. Morabito, "The Internet of Things: A Survey," *Computer Networks*, vol. 54, no. 15, pp. 2787-2805, 2010.
 - [21] S. Tang, D. R. Sheldon, C. M. Eastman, P.-B. Pardis and X. Gao, "A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends," *Automation in Construction*, vol. 101, pp. 127-139, 2019.
 - [22] B. Dave, A. Buda, A. Nurminen and K. Främling, "A framework for integrating BIM and IoT through open standards," *Automation in Construction*, vol. 95, pp. 35-45, 2018.
-

-
- [23] M. Yalcinkaya and V. Singh, “Building Information Modeling (BIM) for Facilities Management – Literature Review and Future Needs,” in *Springer*, Berlin, 2014.
- [24] “People Wiki,” [Online]. Available: https://www.designingbuildings.co.uk/wiki/Facilities_management. [Accessed 03 06 2022].
- [25] “FT maintenance,” [Online]. Available: <https://ftmaintenance.com/maintenance-management/what-is-facility-management/>. [Accessed 04 06 2022].
- [26] D. Bernanrd, M. Frans and W. Roy, “Facilities management: lost, or regained?,” *Facilities*, vol. 30, no. 5/6, pp. 254-261, 2012.
- [27] L. Guillén, J. Antonio, M. Crespo, G. F. Adolfo, G.-P. K. K. Juan Francisco and S. S, “Building Information Modeling as Assest Management Tool,” *IFAC-PapersOnLine*, vol. 49, no. 28, pp. 191-196, 2016.
- [28] I. Errandonea, S. Beltrán and S. Arrizabalaga, “Digital Twin for maintenance: A literature review,” *Computers in Industry*.
- [29] Y. Bouabdallaoui, Z. Lafhaj, P. Yim, L. Ducoulombier and B. Bennadji, “Predictive Maintenance in Building Facilities: A Machine,” *Sensors*, vol. 21, no. 4, p. 1044, 2021.
- [30] V. Villa, B. Naticchia, G. Bruno, K. Aliev, P. Piantanida and D. Antonelli, “IoT Open-Source Architecture for the Maintenance of,” *APPLIED SCIENCES*, vol. 11, no. 12, pp. 1-23, 2021.
- [31] W. Hu, T. Zhang, X. Deng and Z. Lio, “Digital twin: a state-of-the-art review of its enabling technologies, applications and challenges,” *Journal of Intelligent Manufacturing and Special Equipment*, vol. 2, no. 1, pp. 1-34, 2021.
- [32] B. Schleich, N. Anwer, L. Mathieu and S. Wartzack, “Shaping the digital twin for design and production engineering,” *CIRP Annals - Manufacturing Technology*, vol. 66, pp. 141-144, 2017.
- [33] J. Rios, F. Mas, M. Oliva and Hernandez-Matias, “Framework to support the aircraft digital counterpart concept with an industrial design view,” *International Journal of Agile Systems*, vol. 9, no. 3, pp. 212-231, 2016.
-

-
- [34] A. N. Nasaruddin, T. Ito and T. B. Tuan, "Digital twin approach to building information management," in *In Proceedings of the Manufacturing Systems Division Conference*, Stockholm , 2018.
- [35] S. H. Khajavi, N. H. Motlagh, A. Jaribion, L. C. Werner and J. Holmström, "Digital Twin: Vision, Benefits, Boundaries, and Creation for Buildings," *IEEE Access* , vol. 7, pp. 147406 - 147419, 2019.
- [36] X. Xie, Q. Lu, A. K. Parlikad and M. J. Schooling, "Digital Twin Enabled Asset Anomaly Detection for Building Facility Management," *IFAC-PapersOnLine*, vol. 53, no. 3, pp. 380-385, 2020.
- [37] M. Liu, S. Fang, H. Dong and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346-361, 2021.
- [38] M. El Jazzer, M. Piskernik and H. Nassereddine, "Digital Twin in construction: An Empirical Analysis, Proceedings," *EG-ICE 2020 Workshop on Intelligent Computing in Engineering*, , pp. 501-510, August 2020.
- [39] B. R. Barricelli, E. Casiraghi and D. Fogli, "A Survey on Digital Twin: Definitions,," *IEEE Access*, vol. 7, pp. 167653 - 167671, 2019.
- [40] F. Jiang, L. Ma and T. C. K. Broyd, "Digital twin and its implementations in the civil engineering sector," *Automation in Construction*, vol. 130, 2021.
- [41] H. Begic' and M. Galic', "A Systematic Review of Construction 4.0 in the Context of the BIM 4.0 Premise," *Buildings*, vol. 11, no. 337, 2021.
- [42] B. C. Fialho, R. Codinhoto, M. M. Fabricio, J. C. Estrella, C. M. N. Ribeiro, J. M. d. S. Bueno and J. P. D. Torrezan, "Development of a BIM and IoT-Based Smart Lighting Maintenance System Prototype for Universities' FM Sector," *Buildings*, vol. 12, no. 2, 2022.
- [43] B. Dave, A. Buda, A. Nurminen and K. Främling, "A framework for integrating BIM and IoT through open standards," *Automation in Construction*, vol. 95, pp. 35-45, 2018.
-

-
- [44] W. Natephra and A. Motamedi, “Live data visualization of IoT sensors using Augmented Reality and BIM,” in *36th International Symposium on Automation and Robotics in Construction, ISARC*, 2019.
- [45] D. Kazado, M. Kavgić and R. Eskicioglu, “INTEGRATING BUILDING INFORMATION MODELING (BIM) AND SENSOR TECHNOLOGY FOR FACILITY MANAGEMENT,” *Journal of Information Technology in Construction*, vol. 24, pp. 440-458, 2019.
- [46] R. P. Davis, *Advances in Thermal Energy Storage Systems (Second Edition)*, Woodhead Publishing Series in Energy, 2021.
- [47] L. Chamari, E. Petrova and P. Pauwels, “A web-based approach to BMS, BIM and IoT integration: a case study,” in *REHVA 14th HVAC World Congress*, Lima, 2022.
- [48] D. Lau, J. Liu, S. Majumdar, B. Nandy, M. St-Hilaire and C. S. Yang, “A Cloud-Based Approach for Smart Facilities Management,” in *IEEE Conference on Prognostics and Health Management (PHM)*, 2013.
- [49] S. Macfarlane, “University of Cambridge,” [Online]. Available: <https://www.cdabb.cam.ac.uk/sme-blog-ecodomus-digital-journey-sme>. [Accessed 15 06 2022].
- [50] “Siemens,” [Online]. Available: <https://new.siemens.com/global/en/products/buildings/digital-building-lifecycle/ecodomus-software.html>. [Accessed 23 05 2022].
- [51] S. Akro, *Visualizing Monitoring Data through BIM*, Politecnico di Milano: Master in Building Information Modelling, 2020/2021.
- [52] “Forge Autodesk,” [Online]. Available: https://forge.autodesk.com/en/docs/dataviz/v1/developers_guide/introduction/overview/. [Accessed 16 05 2022].
- [53] K. Walmsley, “Connecting Autodesk to the Internet of Things,” Autodesk.
- [54] “Node.js,” [Online]. Available: <https://nodejs.dev/learn>. [Accessed 16 05 2022].
-

- [55] D. Jones, C. Snider, A. Nassehi, J. Yon and B. Hicks, “Characterising the Digital Twin: A systematic literature review,” *CIRP Journal of Manufacturing Science and Technology*, vol. 29, pp. 36-52, 2020.
- [56] “IBM,” [Online]. Available: <https://www.ibm.com/uk-en/topics/facilities-management#:~:text=Facilities%20management%20can%20be%20defined,Capital%20project%20planning%20and%20management>. [Accessed 04 06 2022].
- [57] X. Gao and P. Pishdad-Bozorgi, “BIM-enabled facilities operation and maintenance: A review,” *Advanced Engineering Informatics*, vol. 39, pp. 227-247, 2019.