DOI 10.53136/979122181558019

PROTOTIPAZIONE DI SISTEMI DIGITALI DI SUPPORTO ALLA GESTIONE PRESTAZIONALE DEGLI EDIFICI

ANGELO MASSAFRA[1]
UGO MARIA CORAGLIA[1]
GIORGIA PREDARI[1]
RICCARDO GULLI[1]

[1]*Department of Architecture, Alma Mater Studiorum, University of Bologna, Bologna, Italy*

angelo.massafra2@unibo.it

Abstract
Over the past century, a vast building stock has been constructed in Europe. Today, this heritage may appear unsuitable compared to contemporary needs concerning functional and performance requirements. Managing buildings nowadays requires significant annual expenses, primarily due to their construction obsolescence, functional complexity, and large dimensions, necessitating a holistic approach that integrates economic-financial and technical-functional management within performance-based strategies.
In this scenario, the digital transition, along with a heightened emphasis on performance aspects, allows for a rethinking of building management practices in terms of cost-benefit optimization. Digital management is expected to catalyze a paradigm shift in the field, fostering awareness towards knowledge-oriented decisions. However, the lack of standardized and universally accepted definitions, the vast volume and heterogeneity of data and data sources, and the uncodified approach that typically characterizes built asset management pose obstacles to digitalization in the field. As a result, implementing digital systems today is still resource-intensive, requiring significant technical, human, and financial investments and necessitating high-level multidisciplinary expertise across various domains.
The study introduces BTwin, a software toolkit designed to streamline the creation of digital decision support systems (DDS) for performance-oriented building management. BTwin comprises a Python library that facilitates the integration of building data from various sources, including Building Information Models (BIM), Building Performance Simulation (BPS), and sensors, into graph networks. This integration is reinforced by semantic and ontological principles and is enhanced by the capability to present the data on interactive dashboards. The paper offers a theoretical overview of the toolkit and demonstrates an example of its application.

Keywords: Built Heritage, Decision Support Systems, Knowledge Graph, Digital Twin, BIM.

INTRODUZIONE

Nonostante l'ambiente costruito si trovi al crocevia di molte politiche nazionali ed internazionali volte a gestire gli edifici esistenti in maniera più sostenibile, esso è ancora resistente all'innovazione (European Commission 2023). Questo limite, dovuto alla natura intrinseca del settore delle costruzioni, è probabilmente il principale ostacolo ad una sua radicale trasformazione ecologica e digitale.
A fronte di tale difficoltà, negli ultimi anni il settore AECO (*Architecture, Engineering, Construction and Operation*) ha comunque avviato la sua transizione digitale, spinto dall'avvento di nuove tecnologie digitali all'avanguardia, tra le quali si menzionano l'intelligenza artificiale (IA), il *Building Information Modeling* (BIM), l'*Internet of Things* (IoT), gli strumenti di *Building Performance Simulation* (BPS) e i *Digital Twin* (DT) (de Wilde 2023). In particolare, i DT hanno assunto un ruolo di primo piano nel campo del *building management*, fungendo da paradigma catalizzatore per lo sviluppo di nuovi sistemi digitali di supporto alle decisioni – da qui in poi citati come DDSS (*Digital Decision Support Systems*) – volti a migliorare i processi decisionali nella gestione degli asset edilizi, sia dal punto di vista strategico che operativo (Lu Q., et al. 2022).
A dispetto delle grandi potenzialità dei DT e della rapida crescita scientifica li ha interessati negli ultimi anni, questa tecnologia è ancora ai suoi albori. Prima che avvenga la sua affermazione su vasta scala in contesti di utilizzo rilevanti, si rendono necessari, da un lato, intense attività di prototipazione e sviluppo tecnico-scientifico, dall'altro, grandi sforzi di teorizzazione e concettualizzazione all'interno dell'ambito delle scienze dell'informazione per il patrimonio costruito, un ambito ancora alla ricerca della sua autonomia disciplinare (Boje et al., 2020). La mancanza di definizioni standardizzate e

universalmente valide, la grande mole e l'eterogeneità dei dati e delle fonti di dato e l'approccio (non codificato da regole o modelli scientificamente condivisi) che solitamente caratterizza la gestione degli asset costruiti (Abuimara et al. 2021), ostacolano lo sviluppo organico di tale tecnologia e, più in generale, quella dei DDSS, limitando così la possibilità di trasformare i tanti dati già disponibili in conoscenza di valore utilizzabile dagli addetti del settore.
L'implementazione tecnologica di tali sistemi digitali è, oggi, un'operazione ad alta intensità di risorse tecniche, umane e finanziarie e dai benefici economici e culturali talvolta difficili da comprendere, specialmente nelle fasi di prototipazione degli strumenti. La realizzazione di un DDSS richiede competenze di alto livello fortemente multidisciplinari – tra le varie: fisica delle costruzioni, informatica, elettronica e scienza dei dati – imprescindibili per consentire flussi di dati sicuri, tracciabili, affidabili e, soprattutto, utili e scientificamente validi, da destinarsi ad utenti che non necessariamente hanno competenze scientifiche e/o digitali specifiche.
La comunità scientifica, che, su piccola scala, non sempre dispone delle risorse e delle figure tecniche necessarie per perseguire tali obiettivi, è chiamata, in primis, a teorizzare questi sistemi al fine di chiarire la loro natura e composizione; in secondo luogo, a dimostrare sperimentalmente i benefici tangibili di queste tecnologie ai gestori degli edifici – qui intesi ipoteticamente come i principali beneficiari dei DDSS – e, infine, ad indirizzare teorie e sperimentazioni verso il trasferimento tecnologico delle soluzioni, affinché queste possano essere sviluppate e diffuse anche nella prassi.
Per rispondere a questa esigenza, il presente lavoro presenta BTwin, una libreria software progettata per la prototipazione speditiva, flessibile, modulare e a basso costo di DDSS per la gestione degli edifici. Il toolkit, consistente in una libreria Python, consente di modellare semanticamente diverse tipologie di dato provenienti da varie fonti (quali modelli BIM e BPS oppure dispositivi sensoristici), collegare le informazioni all'interno di database condivisi e, infine, visualizzarle attraverso dei cruscotti interattivi di semplice utilizzo.
Il software si basa su un modello dati basato sulla notazione JavaScript Object Notation (JSON), la cui formattazione è veloce da scrivere e facilmente leggibile ad occhio umano. Sviluppato secondo i principi della programmazione orientata agli oggetti, tale *data model* struttura le informazioni prestazionali degli edifici come nodi e connessioni all'interno di strutture a grafo, sulla base di assiomi e definizioni definiti all'interno di un'ontologia di supporto. Quest'ultima è sviluppata federando alcuni schemi di comprovata importanza nel settore AECO, quali Industry Foundation Classes (IFC) per la rappresentazione degli aspetti spaziali e costruttivi (buildingSMART International 2021), Building Topology Ontology (BOT) (W3C 2021) e la gerarchia di classi di Topologic per gli aspetti topologici (Jabi et al. 2018), l'ontologia di Brick per gli aspetti impiantistici (Balaji et al. 2018) e l'Input Data Format (IDF) di EnergyPlus per gli aspetti energetico-prestazionali (EnergyPlus 2023).
Il toolkit dipende da alcune librerie molto diffuse nella comunità Python, tra cui Pandas per l'analisi dei dati (Pandas 2024), NetworkX per la strutturazione e l'analisi dei grafi (NetworkX 2024), Topologicpy (Topologicpy 2024) per l'analisi topologico-spaziale, Eppy per la simulazione energetica (Eppy 2024), Plotly per la visualizzazione dei dati (Plotly 2024) e Dash per lo sviluppo delle interfacce (Plotly 2024). Questa architettura garantisce scalabilità e modularità delle applicazioni attraverso connettori logici a software terzi. Ad esempio, il connettore con EnergyPlus permette di leggere i risultati delle simulazioni energetiche e di modellarle all'interno del modello dati di BTwin.
L'articolo è organizzato come segue. All'inizio, viene fornito il contesto di riferimento della ricerca. Successivamente, vengono illustrati gli aspetti teorici e ontologici alla base della libreria software. Vengono quindi forniti i dettagli tecnici del toolkit e, infine, descritti alcuni esempi di utilizzo.

STATO DELL'ARTE

Modellazione della conoscenza tramite sistemi a grafo

Quando esprimiamo i nostri pensieri ad altre persone – ad esempio nei luoghi di lavoro – creiamo, consciamente o inconsciamente, delle strutture di conoscenza dette "grafi" (o "*knowledge graphs*" in termini anglosassoni, abbreviato "KG"). Ciò significa identificare degli oggetti specifici ed etichettare le relazioni tra di essi per scomporre un problema, spiegare un concetto o rappresentare un dominio disciplinare, in altre parole modellare delle forme di conoscenza per poterle trasferire a chi ci sta intorno.
I grafi possono essere prodotti (e interpretati) non solo dagli esseri umani, ma anche dai computer. Quelli comprensibili ai computer consistono, tipicamente, in strutture a rete composte da "*nodes*" (o vertici), che denotano entità e soggetti, ed "*edges*" che rappresentano le connessioni tra questi (o relazioni), quindi i predicati.
All'interno dei grafi, sia i nodi che le connessioni possono essere caratterizzati da attribuzioni semantiche e descrittive (Paulheim 2016). Questo significa che l'organizzazione delle informazioni in queste strutture consente, da un lato, di assegnare un significato ontologico agli elementi che compongono un dominio conoscitivo, dall'altro, di descrivere tali elementi attraverso delle proprietà, siano queste qualitative o quantitative, numeriche o testuali. Ne deriva che, una volta che le informazioni sono inquadrate all'interno di un KG, queste possono essere trasferite come conoscenza di valore tramite operazioni di estrazione, aggregazione ed analisi, anche quando queste informazioni formano delle strutture conoscitive molto complesse. Allo stesso modo, un KG può essere segmentato in più grafi per agevolare la comprensione di un dominio di conoscenza tramite la sua scomposizione in sottodomini.
Oltre che in questi aspetti, i KG eccellono nell'analisi di *Big Data* e nell'integrazione di dati che si trovano in ambienti e formati diversi. Inoltre, se accoppiati con le tecnologie per il web semantico, essi possono permettere buona accessibilità alle informazioni tramite il web. Questi sono tutti vantaggi di prioritaria importanza per lo sviluppo di DDSS moderni che hanno fatto sì, nel panorama contemporaneo, che i KG assumessero un ruolo centrale in molte applicazioni digitali anche nel settore AECO (Lygerakis et al. 2022).

Ontologie di riferimento nel settore AECO

Conoscere il significato dei termini utilizzati in un KG è essenziale per organizzare sistematicamente la conoscenza in esso racchiusa e condividerla a terzi in maniera univocamente comprensibile. All'interno di un KG, la descrizione organica delle informazioni viene solitamente ottenuta utilizzando le ontologie. Un'ontologia è una rappresentazione formale, condivisa ed esplicita, della concettualizzazione di un dominio di interesse, che viene formulata per assicurare che tutte le informazioni all'interno del dominio siano chiaramente definite in modo da essere utilizzabili mediante ragionamenti logici.

Negli ultimi anni, con l'esplosione degli strumenti digitali a disposizione del settore delle costruzioni, varie ontologie sono state definite per descrivere il mondo AECO e i suoi sottodomini. Una solida rassegna della letteratura sulle ontologie più utilizzate per lo sviluppo di applicazioni per gli edifici – focalizzate principalmente sul tema energetico (Pritoni et al. 2021). Di seguito vengono brevemente introdotte le sole ontologie utilizzate per costruire l'ontologia federata a supporto dello studio presentato nei successivi paragrafi: IFC, BOT, Topologic, Brick e IDF.

IFC, supportato da buildingSMART International, è lo standard *openBIM* internazionale. Esso consiste in uno schema aperto per la rappresentazione dei dati BIM che vengono scambiati tra i vari stakeholders coinvolti in un progetto di costruzione o gestione di un edificio (buildingSMART International 2021).

BOT è un'ontologia semplificata proposta dal World Wide Web Consortium (W3C) che tratta esclusivamente gli aspetti topologici degli edifici. Attraverso la definizione di vari assiomi e classi di entità, BOT consente la rappresentazione semplificata di componenti fisici e concettuali degli edifici e delle interrelazioni topologiche che sussistono tra essi (W3C 2021).

Topologic è una libreria software per l'analisi spaziale dell'architettura sviluppata dall'Università di Cardiff in collaborazione con l'University College di Londra. Tale libreria si basa su una precisa gerarchia di classi utili a classificare i concetti topologico-spaziali degli edifici e rappresentarli mediante geometrie semplificate (Jabi et al. 2018).

Brick è uno schema *open-source* che standardizza la descrizione semantica degli asset costruiti tramite l'elaborazione di concetti logici, fisici e virtuali, fornendo un approfondimento specifico sul mondo dei sistemi impiantistici. Questo schema viene solitamente utilizzato quando il formato IFC non è in grado di descrivere in modo esaustivo i componenti degli impianti meccanici o elettrici, specialmente quando questi non sono definiti dal punto di vista geometrico (Balaji et al. 2018).

IDF, infine, è il formato dei dati di input di EnergyPlus, software di simulazione energetica per le costruzioni sviluppato dal *Department of Energy* (DOE) degli Stati Uniti. Questo formato consente di descrivere gli elementi e i parametri che caratterizzano il comportamento energetico di un edificio al fine di eseguire delle simulazioni energetiche per il progetto di nuove costruzioni o l'analisi di edifici esistenti.

ONTOLOGIA DI SUPPORTO

Il primo passaggio metodologico seguito per la realizzazione del toolkit consiste nell'identificazione di un'ontologia. Questa ha il compito di fornire una rappresentazione formale degli aspetti prestazionali che riguardano la gestione degli edifici esistenti.

L'ontologia che supporta lo sviluppo BTwin è ottenuta federando le ontologie sopra menzionate, cioè, combinandole in un unico framework, coerente con gli obiettivi della ricerca, mediante l'allineamento dei concetti in esse definiti. Il processo di federazione assicura, da un lato, standardizzazione dell'informazione, dall'altro interoperabilità con sistemi informativi esterni che si appoggiano sulle ontologie federate.

Per rendere utilizzabile tramite ontologia all'interno di KG, BTwin presuppone che un edificio possa essere rappresentato come un grafo composto da nodi (soggetti) e connessioni (predicati). I nodi vengono distinti semanticamente identificandone l'appartenenza a specifiche classi e sottoclassi – definite nell'ontologia – mentre le connessioni tra i nodi vengono contraddistinte tramite il riferimento a specifici tipi di relazioni, anche questi codificati nell'ontologia.

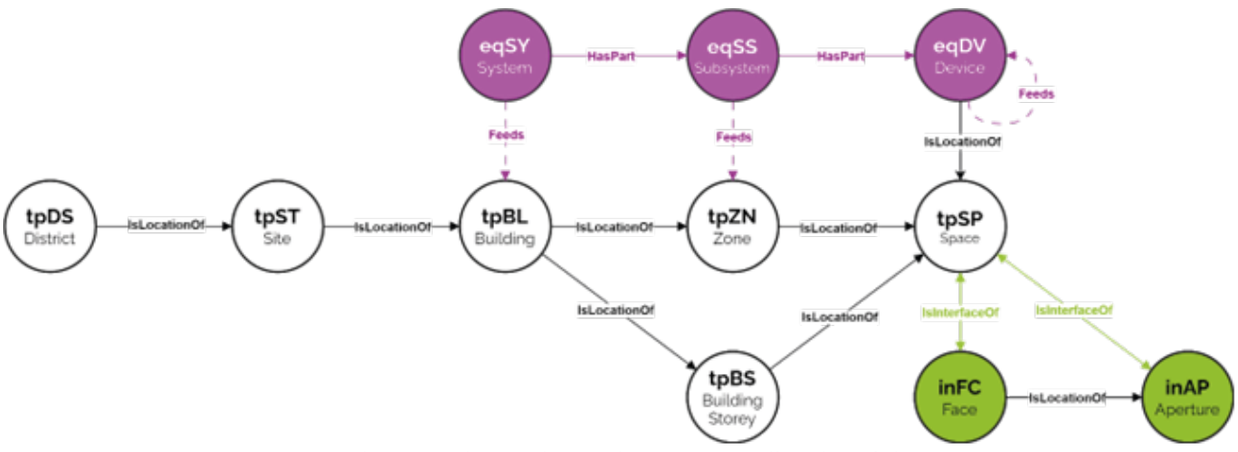

Fig. 1. Classi, sottoclassi e relazioni che compongono l'ontologia federata.

Classificazione delle entità

L'assunzione principale dello studio è che ogni entità (o componente) all'interno di un edificio può essere categorizzata con una classe e una sottoclasse. Le classi utilizzate per la caratterizzazione semantica delle entità – ovvero dei nodi – sono:

- *Topology*: componenti spaziali, ovvero elementi bidimensionali e tridimensionali (fisici o astratti), che compongono un sottoinsieme del mondo (reale o virtuale), mediante la loro esistenza nello spazio. Sono *Topologies*, ad esempio, le zone, gli spazi e i piani che compongono un edificio.
- *Interface*: rappresentazioni astratte degli elementi di partizione – ad esempio pareti, tetti, e solai – e degli elementi in essi contenuti – porte, finestre e aperture – che delimitano i componenti spaziali e li mettono in connessione.
- *Equipment*: sistemi e dispositivi a servizio dei componenti spaziali di un edificio che ne consentono un adeguato utilizzo. In questa classe ricadono i sistemi HVAC (*Heating, Ventilation and Air Conditioning*) e i loro componenti, così come le reti di rilevamento dati e i loro dispositivi.

Come anticipato, tali definizioni sono allineate con quelle delle ontologie presentate al paragrafo 2.2.
In particolare, la classe *Topology* equivale a "*IfcSpatialStructureElement*" di IFC, "*Location*" di Brick e "*Zones*" di BOT e IDF. La classe *Interface* è allineata alla classe "*Interface*" di BOT, "*Face*" di Topologic e "*Surface*" di IDF. Infine, la classe *Equipment* richiama l'omonima classe descritta in Brick.
Ogni classe è poi caratterizzata dalle sue sottoclassi.
Una *Topology* può consistere in un "*District*", "*Site*", "*Building*", "*BuildingStorey*", "*Zone*" e "*Space*". Questi elementi, che formano la gerarchia spaziale di BTwin, fanno essenzialmente riferimento alla gerarchia spaziale di IFC con l'aggiunta della sottoclasse *District*, definibile come un'area geografica o una regione all'interno di un contesto amministrativo, politico o funzionale, la cui identificazione è utile alla gestione di un patrimonio immobiliare. Per i patrimoni universitari, ad esempio, la definizione di *Disctrict* corrisponde con quella di campus.
Le *Interfaces* si distinguono invece in "*Face*" e "*Aperture*", richiamando le omonime classi di Topologic. Le *Faces* comprendono le interfacce verticali, orizzontali o inclinate che suddividono spazi (o zone) da altri spazi (o zone) o dall'esterno, quindi pareti, coperture e solai. Le *Apertures* si riferiscono invece alle aperture all'interno delle *Faces* che consentono il passaggio di persone, aria o luce, quindi porte, finestre e varchi.
Le entità *Equipment*, infine, possono essere classificate come "*System*", "*Subsystem*" oppure "*Device*". I *Systems* sono da intendersi come insiemi di circuiti impiantistici interconnessi che svolgono funzioni specifiche all'interno di un edificio. Un esempio di *System* è il sistema HVAC di un edificio. I *Subsystems* sono invece definibili come insiemi, più ristretti, di componenti interconnessi progettati per svolgere funzioni specifiche all'interno di una zona o di uno spazio di un edificio. Un esempio è il circuito di riscaldamento di una zona. I *Devices*, invece, si riferiscono ai dispositivi specializzati che compongono un *Subsystem* che, relazionandosi con altri dispositivi, permettono lo svolgimento delle funzioni per cui tale *Subsystem* è stato progettato. Esempi di *Devices* che compongono un circuito di riscaldamento sono i radiatori, le pompe di calore o le caldaie (ma non solo).

Tipizzazione delle relazioni

Sulla base della loro appartenenza alle classi e sottoclassi menzionate, le entità (nodi) possono essere relazionate da diversi tipi di connessioni. Si menzionano di seguito i principali tipi di relazioni supportati da BTwin, mantenendo anche in questo caso i termini in inglese per favorire la leggibilità degli allineamenti concettuali effettuati con le ontologie di riferimento.
La relazione "*HasLocation*" identifica se un'istanza è contenuta spazialmente all'interno di un'altra istanza; al contrario, "*IsLocationOf*" individua se un'istanza contiene un'altra istanza. Questo tipo di relazione può essere applicata tra istanze *Topology* (ad esempio, "uno spazio è contenuto in un piano"), tra istanze *Interface* ("una finestra è contenuta in una parete") oppure tra istanze *Device* e istanze *Topology* ("un boiler è collocato in uno spazio").
Le istanze *Interface* e le *Topology* possono essere invece collegate attraverso le relazioni "*IsInterface*Of" o "*HasInterface*". Queste relazioni identificano principalmente l'adiacenza tra una *Face* (o *Aperture*) e gli elementi spaziali da essa delimitati ("un tetto è un'interfaccia di uno spazio"). Queste due relazioni sono particolarmente utili per identificare l'adiacenza o il passaggio tra due spazi all'interno delle simulazioni di prestazione. Ad esempio, se una parete funge da interfaccia tra due spazi, significa che tali spazi sono adiacenti. Oppure, se una porta funge da interfaccia di due spazi, significa che tali spazi sono collegati tramite un passaggio.
Due tipi aggiuntivi identificano poi i collegamenti tra le istanze *Equipment* e tra le istanze *Equipment* e *Topology*. Le relazioni "*HasPart*" e "*IsPartOf*" consentono di navigare nella gerarchia *System-Subsystem-Device*. Ad esempio, un radiatore fa parte di un circuito di riscaldamento, che a sua volta fa parte del sistema HVAC di un edificio. D'altra parte, le relazioni "*Feed*" e "*IsFedBy*" consentono di connettere *Systems* e *Subsystems* agli elementi spaziali. Per esempio, un circuito di riscaldamento alimenta la zona di un edificio. Inoltre, tali relazioni consentono di connettere *Devices* diversi per identificare il passaggio di fluidi, dati o energia tra essi (es., "una caldaia alimenta un radiatore").

Attribuzione delle informazioni

Oltre che di una classificazione semantica e gerarchica, le entità modellate all'interno di BTwin possono essere dotate di dati e metadati che descrivono, qualitativamente o quantitativamente, le entità stesse. In particolare, tali dati possono essere attribuiti ai nodi come:

- *Points*: indicano unità isolate di informazioni (o punti dato) che riportano osservazioni effettuate in un istante specifico. Questi denotano, di fatto, i valori di determinati parametri registrati da un dispositivo, da un sistema o da un'entità spaziale. Esempi di Points includono: una misurazione di temperatura effettuata da un sensore a una

certa ora di un certo giorno, una registrazione del consumo di energia elettrica effettuata da un contatore in un certo mese, oppure un valore di fabbisogno energetico di una certa zona calcolato da una simulazione energetica.

- *PropertySets*, abbreviati "*PSet*", si riferiscono a gruppi di proprietà e attributi di un'istanza organizzate per tematica, disciplino o tipologia di dato. Esempi di *PSet* possono essere i requisiti termici degli spazi o le caratteristiche termiche di un solaio.
- *KeyPerformanceIndicatorSets*, o *KPISet*, comprendono insiemi di metriche e indicatori utili a valutare le prestazioni dei componenti spaziali di un edificio durante un determinato periodo di tempo. Un esempio di *KPISet* utile alla gestione prestazionale di un asset può consistere nel raggruppamento di tutte le metriche relative alla stima dei fabbisogni energetici di un certo spazio durante un certo anno.

Nei grafi di BTwin, *Points*, *PSet*s e i *KPISets* vengono anch'essi rappresentati come nodi. Questi possono essere collegati alle altre entità attraverso tipi di relazione specifici.
Nel caso di dati spaziali simulati, le relazioni "*IsPointOf*" e "*HasPoint*" consentono di collegare i *Points* alle istanze *Topologies* (es., "un certo valore di energia è richiesto per riscaldare una certa zona ad una certa ora") e alle istanze *Equipment* ("un certo valore di umidità è registrato ad una certa ora da un certo sensore").
I *PSet* possono stabilire collegamenti con qualsiasi istanza attraverso le relazioni "*IsPSetOf*" e "*HasPSet*". Diversamente, i *KPISet* possono collegarsi ai soli elementi spaziali, attraverso le relazioni "*IsKPISetOf*" e "*HasKPISet*". Infine, *PSets* e *KPISets* sono associabili, rispettivamente, a proprietà e indicatori attraverso le relazioni "*HasProperties*" e "*HasKPIs*".

MODELLAZIONE DEI DATI

Operativamente, la modellazione dei dati in BTwin è supportata da un modello dati specifico. Questo si basa su una sintassi specifica che utilizza oggetti JSON per rappresentare le entità e le loro relazioni. Il vantaggio di questa notazione è quello di rendere i dati facilmente scrivibili e leggibili dall'uomo – oltre che dalle macchine – soprattutto se confrontata con formati quali EXPRESS di IFC. Grazie alla sua estensibilità e adattabilità, JSON consente la serializzazione dei dati all'interno di dizionari, tra loro relazionabili per creare delle strutture a grafo. Inoltre, grazie alla sua leggerezza, questo formato si dimostra particolarmente adatto allo scambio di dati in ambienti web. L'assunzione fondamentale per la rappresentazione delle informazioni all'interno del *data model* è che ogni elemento edilizio – sia esso una *Topology*, un'*Interface* o un *Equipment* – deve corrispondere a un oggetto JSON dotato di un proprio codice identificativo univoco (anche detto "*Unique Identifier*", UID). Questo oggetto JSON ha la forma di un dizionario, dotato almeno di un proprio UID, dell'identificatore univoco della sottoclasse a cui appartiene (vedi Figura 1) e, eventualmente, dell'indicazione delle relazioni che tale oggetto di istanza ha con le altre istanze. Tutte queste informazioni sono riportate nel dizionario all'interno di coppie di *key* and *values*. Rispettando una specifica sintassi per la modellazione dei dati, il modello dati JSON può essere letto da BTwin e rapidamente trasformato in un KG utilizzando delle funzioni specifiche.
Di seguito viene presentata la descrizione sintattica di alcune entità categorizzabili mediante le classi precedentemente illustrate per chiarire come avviene il processo di generazione del KG.

Rappresentazione delle istanze

La Fig. 2 mostra, a sinistra, come viene rappresentata un'istanza di *Topology* all'interno del formato JSON. All'interno dell'oggetto JSON, la chiave "UID" contiene l'identificatore univoco, <space1>, dello spazio in questione, mentre l'identificatore "tpSP", che rappresenta la sottoclasse dello spazio, è assegnato alla chiave "*Class*". Le relazioni dello spazio con gli altri oggetti sono definite nella chiave "*rl_Relationships*". Il valore corrispondente a quest'ultima chiave è un ulteriore dizionario, dove ogni tipo di relazione funge da chiave e l'UID dell'oggetto correlato funge da valore. Ad esempio, il posizionamento dello <spazio1> all'interno del piano <pianoX1A> è stabilito dalla chiave "*HasLocation_tpBS*". Similmente, l'associazione dello spazio con le zone energetiche <energyzoneA> e <firezoneA> è indicata dalla chiave "*HasLocation_tpZN*".

a) Space

```
{
  "Class": "tpSP",
  "UID": "space1",
  "rl_Relationships": {
                    "HasLocation_tpBS":
["storeyX1A"],
    "HasLocation_tpZN":[
      "energyzoneA",
      "firezoneA"],
    "IsAdjacentTo_tpSP": ["space3"],
    "HasPassageTo_tpSP": ["space2"]
  }}
```

b) Point

```
{
  "Class": "ptSN",
  "UID": "point110",
  "ts_Timestamp": "202211151000",
  "rl_Relationships": {
                                "IsPointOf_eqDV":
["temperatureSensor1"]
  },
  "vl_Value":{
    "quantity":"Temperature_C",
    "value":22.79
  }}
```

c) KPISet

```
{
  "UID": "OperationalEnergy",
  "Class": "ksKS",
  "rl_Relationships": {
    "IsKPISetOf_tpBL":
["buildingX1"]
  },
  "ks_HasKPIs": {
    "k_HeatingEnergyDemand": {
      "value": 18.85,
      "quantity": "Energy_kWh"
    },
    "k_CoolingEnergyDemand": {
      "value": 5.44,
      "quantity": "Energy_kWh"
    },
  },
  "ts_Timestamp": {
    "from": 202201010100,
    "to": 202212312400
  }
}
```

Fig. 2. Rappresentazione di uno spazio, un punto e un insieme di indicatori nel formato JSON.

Rappresentazione di dati, indicatori e proprietà

La Fig. 2 illustra, al centro, la formattazione di un *Point* secondo la sintassi JSON. La chiave "UID" contiene l'identificatore univoco del punto, <punto110>, mentre l'identificatore di classe "ptSN", assegnato alla chiave "*Class*", rappresenta la sottoclasse del punto (dove "pt"sta per "*Point*" e "SN" per "s*ensor*"). Le relazioni tra il punto e le altre istanze sono definite nella chiave "*rl_Relationships*". In particolare, il <punto110> è registrato dal dispositivo <temperatureSensor1>, che è un sensore di temperatura. Questa relazione è stabilita dalla chiave "*IsPointOf_eqDV*".

L'oggetto JSON in Figura 2b ha due chiavi aggiuntive. La chiave '*ts_Timestamp*' si riferisce all'orario in cui il valore è stato registrato dal punto, espresso nel formato YYMMDDhhmm. La chiave '*vl_Value*' riporta invece il valore e la grandezza del parametro misurato. Nell'esempio in figura, si visualizza un valore di temperatura pari a 22.79°C registrato dal <temperatureSensor1> alle ore 10:00 del 15 novembre 2022.

Infine, la Fig. 2 fornisce, a destra, un esempio di *KPISet* in formato JSON. Questo *KPISet* determina le prestazioni dell'edificio <buildingX1> in termini di consumi energetici. All'interno di questo oggetto JSON, la chiave "UID" contiene l'identificatore unico, <OperationalEnergy>, per il *KPISet* in questione. Le relazioni del *KPISet* sono definite nella chiave "*rl_Relationships*". In sostanza, la relazione "*IsKPISetOf_tpBL*" esplicita che <OperationalEnergy> è un *KPISet* dell'edificio <buildingX1>. La chiave "*ks_HasKPIs*" si riferisce invece alle relazioni del *KPISet* con gli indicatori che lo compongono, "*kp_HeatingEnergyDemand*" e "*kp_CoolingEnergyDemand*". Infine, la chiave "*ts_Timestamp*" fornisce un'indicazione del periodo a cui si riferisce il *KPISet*.

La notazione dei *PSets* è molto simile a quella dei *KPISets*, con la differenza che i *PSets* non dispongono di una chiave "*ts_Timestamp*" e che la chiave "*ks_HasKPIs*" è sostituita dalla chiave "*ps_HasProperties*".

APPARATO STRUMENTALE

Dal punto di vista strumentale, BTwin è una libreria Python interconnessa con altre librerie di riferimento nel settore, ben documentate e supportate da vaste comunità di sviluppatori. In estrema sintesi, sfruttando i concetti ontologici esposti e modellando i dati seguendo la sintassi JSON proposta, BTwin consente una rapida e semplice prototipazione di DDSS per gli edifici.

Come già spiegato, il software utilizza un formato di dati basato sulla notazione JSON. Questo formato può essere analizzato e convertito in reti a grafo utilizzando delle funzioni dedicate. Più precisamente, per costruire la struttura dei grafi viene utilizzata la libreria NetworkX (NetworkX 2024). Si tratta di una libreria Python che fornisce un insieme di strumenti per la creazione, la manipolazione e lo studio di reti complesse.

Utilizzando la cosiddetta logica dei "connettori", lo strumento si integra organicamente con librerie e formati esterni, promuovendo scalabilità e modularità nello sviluppo dei DDSS. Ad esempio, il connettore con Topologicpy (Topologicpy 2024) consente di utilizzare modelli realizzati in Topologic per modellare le entità spaziali direttamente all'interno dei grafi. Allo stesso modo, il connettore con EnergyPlus, realizzato integrando le API della libreria Eppy (Eppy 2024), facilita il collegamento dei risultati delle simulazioni energetiche e gli elementi spaziali il cui comportamento è simulato. Ulteriori connettori permettono di interfacciarsi, seppure in maniera semplificata, con Autodesk Revit, utilizzando le API fornite da Autodesk. Questi connettori sono utili per estrarre le proprietà degli elementi spaziali all'interno dei modelli BIM e associarle ai nodi del grafo. Inoltre, è stata testata la compatibilità con Arduino per la modellazione di dati sensoristici e sono state effettuate delle sperimentazioni preliminari con le API di ChatGPT per consentire l'estrazione di informazioni dai grafi mediante prompt testuali (Saka et al. 2024). L'architettura modulare della libreria consentirà, in futuro, lo sviluppo di ulteriori connettori con formati dati più complessi, quali IFC o gbXML.

Una volta che i dati sono interconnessi all'interno del grafo – con o senza l'utilizzo di connettori – il toolkit abilita varie funzioni per l'interrogazione, l''aggregazione, l'elaborazione e la visualizzazione dei dati custoditi all'interno del KG. In particolare, il modulo di visualizzazione si appoggia sulle librerie Plotly e Dash (dash Plotly 2024) per la realizzazione di cruscotti web prototipali. Ne risultato delle micro-applicazioni web che possono essere facilmente pubblicate in Internet utilizzando soluzioni quali Docker e AWS (Betti et al. 2022).

ESEMPI APPLICATIVI

In questa sezione viene presentata l'applicazione metodologica del software esemplificando alcuni possibili *workflow* di utilizzo. A fine di maggiore chiarezza, le applicazioni presentate sono testate su un caso di studio fittizio, consistente in un semplice edificio di un piano, costituito da tre spazi cellulari aventi dimensioni in pianta pari a 5x5 metri e dotati di un'altezza lorda pari a 3 metri. Nello specifico, dopo aver modellato la gerarchia delle istanze di *Topology* ed *Equipment* che compongono il caso di studio, viene fornita un'applicazione di esempio finalizzata al monitoraggio del comfort termico degli spazi.

Modellazione della gerarchia spaziale

La gerarchia spaziale del caso studio viene creata in Topologic e poi importata in un sistema a grafo utilizzando i connettori BTwin. Gli elementi che compongono la gerarchia spaziale del caso di studio sono identificati nella Fig. 3° con dei nodi colorati. In questo esempio, l'istanza di distretto <disctrictA> contiene l'istanza di sito <siteX>. Un'istanza di edificio, <edificioX1>, è collocato nel <siteX>. Questo edificio contiene un *BuildingStorey*, <storeyX1A>, in cui sono collocati tre spazi, <space1>, <space2> e <space3>. Tutte le connessioni all'interno della gerarchia spaziale sono specificate dalle relazioni "*HasLocation*" e "*IsLocationOf*".

Modellazione della gerarchia impiantistica

L'esempio della Fig. 3° è completato dagli elementi di impianto, mostrate nella Fig 3b. Nella Fig. 3b, viene illustrato il sistema di sensing di cui è dotato il caso studio, <sensingSystemX>. Questo è suddiviso in tre sottosistemi: <sensingNetwork1>, < sensingNetwork2> e < sensingNetwork3>. Ognuno di questi tre sottosistemi è composto da un sensore di temperatura (<temperatureSensorX>), uno di umidità (<humiditySensorX>) e uno di illuminazione (<illuminanceSensorX>). Questi sensori sono posizionati rispettivamente nello spazio corrispondente che, secondo la nomenclatura stabilita, è lo spazio <spaceX>. I collegamenti all'interno della gerarchia degli impianti sono dati dalla relazione "*IsPartOf*", il posizionamento dei sensori all'interno degli spazi è invece dato dalla relazione "*HasLocation*". Si veda la Fig. 4 per aggiore chiarezza.

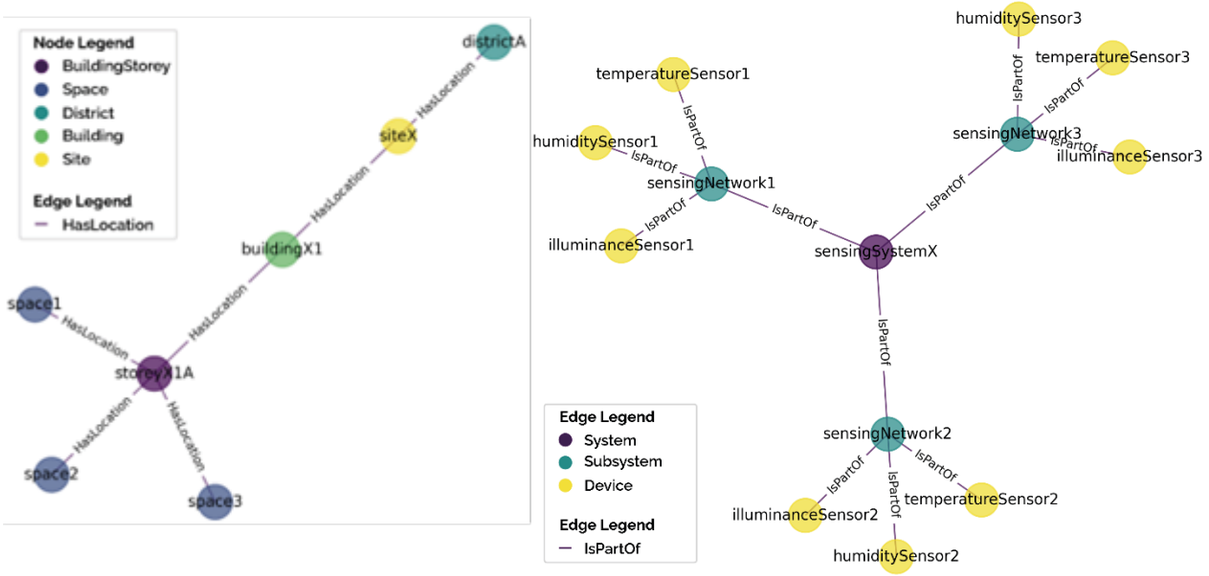

Fig. 3. Gerarchia spaziale (a sinistra) e impiantistica (a destra) del caso studio all'interno di un KG.

Sviluppo dell'applicazione

Di seguito viene presentato il processo seguito per la realizzazione di un applicativo prototipale finalizzato al monitoraggio del comfort termico all'interno degli spazi che compongono il caso studio presentato. Il risultato è un'interfaccia che permette di valutare il numero di ore di discomfort di temperatura, umidità e illuminamento all'interno dei tre spazi, utilizzando delle misurazioni effettuate durante una giornata invernale. I dati utilizzati per la prototipazione sono anch'essi fittizi, essendo l'articolo destinato alla dimostrazione metodologica del toolkit presentato, più che all'analisi specifica di un caso studio realmente esistente.

Dal punto di vista metodologico, questo processo può essere ripetuto per utilizzi informativi diversi da quello qui illustrato, quali analisi dei fabbisogni energetici degli spazi, rilevamento della qualità dell'aria, stima della sicurezza, valutazione delle condizioni di utilizzo degli spazi, e così via. Future attività di ricerca riguarderanno il test dello strumento su casi studio reali al fine di stimarne limiti e potenzialità metodologici e strumentali.

Si riportano i passaggi principali seguiti per l'implementazione dell'applicazione in Fig. 5:

1. Innanzitutto, si è creata gerarchia spaziale dell'edificio all'interno di un KG (Fig. 3°);
0. Successivamente, sono stati aggiunti al grafo gli elementi impiantistici (Fig. 3b), e sono state stabilite le relazioni tra questi e gli elementi spaziali;
0. Una volta identificati tutti gli elementi spaziali e impiantistici, sono state modellate le proprietà degli elementi spaziali. Per ogni spazio, si sono aggiunte informazioni relative a valori target di comfort ambientale, definite nel KG come proprietà degli spazi raggruppate all'interno di *Psets* denominati "*ComfortRequirements*". I valori di queste proprietà sono stati settati, a titolo esemplificativo, pari un valore minimo di 20°C per la temperatura, un intervallo tra il 30% e il 60% per l'umidità e un valore minimo di 250 lux per l'illuminamento;
0. I dati relativi alle misurazioni effettuate dai sensori – in maniera fittizia – sono stati formattati in JSON e modellati nel grafo come *Points* collegati ai dispositivi sensoristici che compongono il sistema di rilevamento dell'edificio;
0. Tali informazioni sono estratte dal grafo per calcolare, spazio per spazio, gli indicatori di comfort ambientale interno. Tali KPI indicano, sempre a titolo esemplificativo, il numero di ore in cui i valori target di comfort (settati al punto 3) non sono soddisfatti. Per ogni spazio, tali KPI vengono raggruppati in un *KPISet*, poi collegato al nodo che rappresenta lo spazio stesso nel grafo;
0. Attraverso Dash, viene creato un cruscotto per visualizzare i KPI sulle geometrie del modello 3D dell'edificio, prima realizzato in Topologic.

L'esecuzione di questi passaggi risulta nel grafo in Fig. 4 e nella dashboard in Fig. 5.

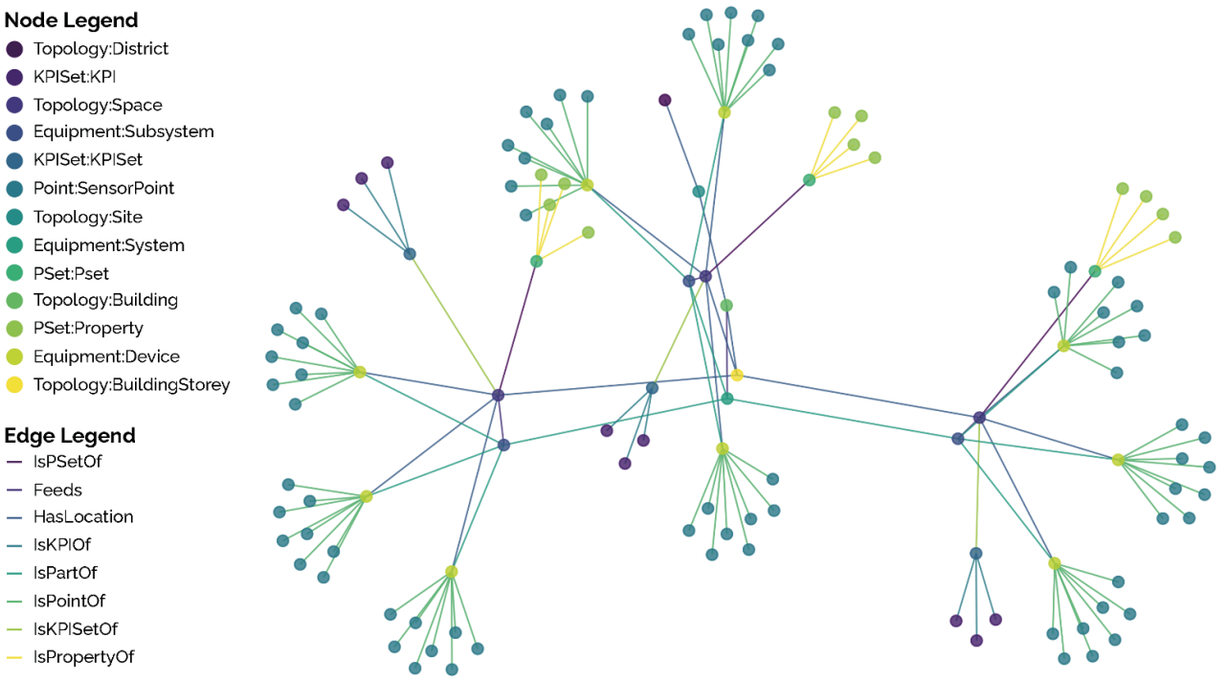

Fig. 4. Rete a grafo per l'organizzazione delle informazioni relative al caso studio presentato.

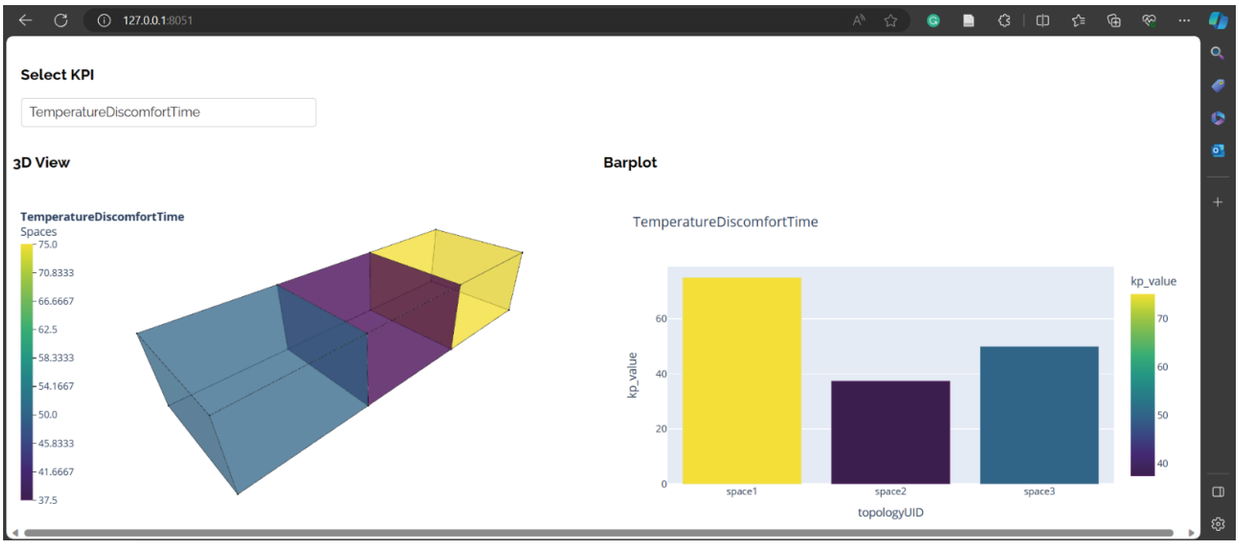

Fig. 5. Dashboard interattiva per la visualizzazione dei KPI per l'analisi prestazionale del caso studio presentato.

CONCLUSIONI

Seppur contrastato da alcune limitazioni intrinseche alla sua stessa natura, il settore AECO sta attualmente vivendo la sua transizione digitale. In riferimento al tema della gestione del patrimonio costruito, tale trasformazione si sta manifestando oggigiorno in una maggiore necessità di disporre di strumenti digitali a supporto dei processi decisionali.
Nonostante i tanti progressi tecnologici e l'esplosione di nuove tecnologie e paradigmi per la gestione negli ultimi anni, quali i gemelli digitali, sono ancora necessari notevoli investimenti tecnici ed economici per l'implementazione di DDSS.
Per aiutare a soddisfare questa crescente necessità, questo articolo ha presentato Btwin, una libreria software progettata per fornire un toolkit semplice, flessibile e modulare per la prototipazione *cost-effective* di DDSS per l'analisi e la gestione delle prestazioni degli edifici esistenti. Il toolkit è composto da diverse funzioni Python che permettono di connettere i dati prestazionali degli edifici – anche quando provenienti da fonti diverse come modelli BIM, BPS o sensori – all'interno di reti a grafo e visualizzarli attraverso cruscotti interattivi di facile interrogazione. Tale integrazione si basa su precise regole semantiche ed ontologiche definite a monte dello sviluppo dello strumento.
Il toolkit presentato mira, quindi, a facilitare la prototipazione di DDSS per la gestione degli edifici in modo semplice e standardizzato. Ciò consente agli sviluppatori di realizzare rapidamente dei prototipi che, basandosi su *dashboard*, possono essere testati dagli utenti anche in forma prototipale in modo da suggerire dei feedback per lo sviluppo delle soluzioni

definitive. Inoltre, l'utilizzo di una sintassi di facile lettura rende lo strumento utile anche per chi si approccia per le prime volte ai campi del BIM, dei DT e dei KG tramite programmazione in codice.
All'interno dell'articolo si è prima fornito il *background* teorico che supporta Btwin, descrivendo gli assiomi, i concetti e le classi su cui questo software si basa. Poi, è stata mostrata una dimostrazione di utilizzo dello strumento su un caso di studio fittizio al fine di mostrare alcuni esempi di utilizzo. Il documento, nel suo insieme, ha valore di *whitepaper* del toolkit che, essendo ai suoi esordi, è attualmente in fase di sviluppo. Gli sviluppi futuri prevedono la creazione di ulteriori funzionalità e connettori. Questi saranno sviluppati allineando il formato dati JSON di Btwin con strutture di dati più complesse, quali IFC e RDF (Resource Description Framework), in modo da assicurare la compatibilità della libreria con gli standard più affermati di scambio dati BIM e web. Ulteriori implementazioni si concentreranno sulla dimostrazione di applicazioni su casi di studio reali più complessi di quello qui mostrato dal punto di vista dimensionale, funzionale e topologico.

RINGRAZIAMENTI
Questa ricerca è stata finanziata dal progetto BeTwin ("Building Digital Twins for Built Heritage Performance-Based Management"), nell'ambito del programma AlmaValue 2023 dell'Università di Bologna e dal Ministero delle Imprese e del Made in Italy, finanziato dal programma NextGenerationEU.
Inoltre, la pubblicazione è stata realizzata da un ricercatore con contratto di ricerca cofinanziato dall'Unione Europea – PON Ricerca e Innovazione 2014-2020 ai sensi dell'Art. 24, comma 3, lett. A), della Legge 30 dicembre 2010, n. 240 e successive modifiche e del D.M. 10 agosto 2021, n. 1062. 1062 del 10 agosto 2021.

BIBLIOGRAFIA
European Commission, 2023, *Built4People*. Accessed Jan. 15, 2024. [Online]. Available: https://build-up.ec.europa.eu/en/resources-and-tools/links/built4people-b4p
de Wilde P., 2023, *Building performance simulation in the brave new world of artificial intelligence and digital twins: A systematic review.* Energy and Buildings, 292: 113171, 2023. Doi: 10.1016/j.enbuild.2023.113171.
Lu Q., et al., 2022, *Digital twins in the built environment: fundamentals, principles and applications*. ICE Publishing, London.
Boje C., et al., 2020, *Towards a semantic Construction Digital Twin: Directions for future research.* Automation in Construction, 114: 103179. Doi: 10.1016/j.autcon.2020.103179.
Abuimara T., et al., 2021, *Current state and future challenges in building management: Practitioner interviews and a literature review.* Journal of Building Engineering, 41: 102803. Doi: 10.1016/j.jobe.2021.102803.
buildingSMART International, 2021, *International Foundation Classes (IFC)*. Accessed: Aug. 29, 2021. [Online]. Available: https://www.buildingsmartitalia.org/standard/standard-bs/industry-foundation-classes-ifc/
W3C, 2021, *Building Topology Ontology*. Accessed Jan. 15, 2024. [Online]. Available: https://w3c-lbd-cg.github.io/bot/
Jabi W., et al., 2018, *Topologic – A toolkit for spatial and topological modelling*. Atti di convegno di eCAADe 2018: Computing for a better tomorrow, Łódź, Poland, pp. 449–458. Doi: 10.52842/conf.ecaade.2018.2.449.
Balaji B. et al., 2018, *Brick : Metadata schema for portable smart building applications.* Appl. Energy, 226: 1273–1292, 2018. Doi: 10.1016/j.apenergy.2018.02.091.
EnergyPlus, 2023. Accessed: May 08, 2023. [Online]. Available: https://energyplus.net/
Pandas, 2024, *Python Data Analysis Library*. Accessed: Jan. 12, 2024. [Online]. Available: https://pandas.pydata.org/
NetworkX, 2024, *NetworkX Documentation*. Accessed: Jan. 12, 2024. [Online]. Available: https://networkx.org/
Topologicpy, 2024, *API documentation*. Accessed: Jan. 12, 2024. [Online]. Available: https://topologic.app/topologicpy_doc/topologic_pdoc/index.html
Eppy, 2024. *Welcome to eppy's Documentation!* Accessed: Jan. 12, 2024. [Online]. Available: https://eppy.readthedocs.io/en/latest/index.html
Plotly, 2024, *Plotly: Low-Code Data App Development*. Accessed: Jan. 12, 2024. [Online]. Available: https://plotly.com/
dash Plotly, 2024, *Dash Documentation & User-Guide*, 2024 Accessed: Jan. 12, 2024. [Online]. Available: https://dash.plotly.com/
Paulheim H., 2016, *Knowledge graph refinement: A survey of approaches and evaluation methods*. Semantic Web, 8(3): 489–508. doi: 10.3233/SW-160218.
Lygerakis F., et al., 2022, *Knowledge Graphs' Ontologies and Applications for Energy Efficiency in Buildings: A Review.* Energies,15(20): 7520. doi: 10.3390/en15207520.
Pritoni T. et al., 2021, *Metadata Schemas and Ontologies for Building Energy Applications: A Critical Review and Use Case Analysis*. Energies, 14(7): 2024, 2021. doi: 10.3390/en14072024.
Saka A. et al., 2024, *GPT models in construction industry: Opportunities, limitations, and a use case validation*. Dev. Built Environ., 17: 100300. doi: 10.1016/j.dibe.2023.100300.
Betti G., et al., 2022, *CBE Clima Tool: a free and open-source web application for climate analysis tailored to sustainable building design*. doi: 10.48550/ARXIV.2212.04609.