Smart City Digital Twin Framework for Semantic Indexing and Dynamic Modelling of 3D Buildings

Marco Fanfani[1(✉)], Federico Galli[1], Chiara Garau[2], Paolo Nesi[1], and Gianni Pantaleo[1]

[1] DISIT Lab, DINFO, University of Florence, Florence, Italy
{marco.fanfani,federico.galli,paolo.nesi,
gianni.pantaleo}@unifi.it
[2] DICAAR, University of Cagliari, Cagliari, Italy
cgarau@unica.it

Abstract. Smart City Digital Twins (SCDT), including 3D representations of the urban environment, have gained ever-increasing attention from academic, public, and private sectors in the last years. Being faithful, dynamic, three-dimensional replicas of the cities, SCDTs realize concrete decision support systems to facilitate the work of urban planners and decision makers. Over the years, several municipalities have succeeded in producing 3D city maps, particularly useful in managing and planning the development of the built infrastructure. However, in most cases such representations are static replicas without any semantic enrichment and offer very limited interactive visualization and manipulation, thus inadequate to support what-if analysis. In this paper, the 3D city modelling and visualization framework embedded into the Snap4City Digital Twin platform is proposed. Each building is represented with multiple 3D versions, augmented with semantic information, and modelled as an individual entity by exploiting the KM4City ontology. Web-based 3D dashboards are used to present the urban twin and allow users to inspect, control and modify in real-time the 3D virtual environment in order to evaluate different scenarios for development of the urban spaces.

Keywords: 3D Building · Knowledge Base · Ontology · Smart City · Digital Twin

1 Introduction

In recent years, Smart City Digital Twins (SCDTs) have evolved from static 3D city visualization tools to dynamic, data-driven decision support systems that integrate heterogeneous urban data, real-time streams, and interactive analytics into a unified digital environment [1–3]. Within SCDTs, buildings are central because they connect to mobility, energy, environment, and service organization, serving as primary units for scenario analysis and policy testing and representing the physical backbone through which many urban processes become observable, measurable, and manageable [4, 5].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026 Workshops, LNCS 16766, pp. 21–34, 2027.
https://doi.org/10.1007/978-3-032-30539-8_2

However, many existing 3D city models remain largely visual, with limited semantic enrichment and interactive capabilities, and are therefore still used mainly as descriptive representations rather than as operational components of urban intelligence systems, hindering scalable decision support at the building level. At the same time, the growing availability of geospatial data, IoT information, and multi-source urban datasets makes it increasingly necessary to move beyond purely geometric modelling approaches and toward solutions capable of linking 3D objects with structured knowledge and dynamic attributes. These gaps motivate the need for a framework capable of combining semantic indexing, dynamic 3D modelling, and interactive visualization in a coherent and scalable way, so that buildings are no longer treated as static graphic objects, but as semantically enriched entities that can be queried, updated, compared across temporal states, and manipulated to explore alternative urban scenarios. In this context, the objective of this article is to investigate the following key research question: how can 3D building representations within SCDTs be semantically enriched and dynamically managed in order to support scalable, interactive, and data-driven urban decision-making? To address this question, the paper proposes a framework embedded in the Snap4City Digital Twin platform, where buildings are modelled as semantic entities through the KM4City ontology, associated with multiple 3D representations, and made accessible through interactive web-based dashboards for real-time inspection and scenario analysis. More specifically, the proposed approach aims to bridge the gap between semantic interoperability, large-scale building data management, and user-oriented 3D interaction, thereby supporting not only the visualization of urban space but also the exploration of possible transformations and what-if scenarios. To achieve this, the paper is organized as follows: Section 2 presents the semantic building modelling approach and the entity creation process within the Snap4City platform. Section 3 describes the dynamic and interactive 3D visualization framework and its main functionalities, while in Sect. 4 validation activities are reported to demonstrate the concrete applicability of the proposed framework. Finally, Sect. 5 draws the conclusions and discusses possible future developments.

2 Semantic Building Modelling and Entity Creation

In this section, the Snap4City smart city digital twin framework (https://snap4city.org) [2] is presented, focusing on building representation and management. First, the Snap4City platform is briefly introduced, illustrating its main functionalities. Then details on the solution used to model building entities into Snap4City are discussed, presenting their semantic indexing according to the KM4City ontology [6, 7] and the defined data model. Finally, the ingestion process implemented with processing logic embedded into the platform is described.

2.1 Snap4City Platform

Snap4City is an open-source platform designed to realize and implement Smart Cities Digital Twin (SCDT) in order to provide a decision support system helping decision makers in operational monitoring activities and in strategic and tactical planning of the urban environment. The platform is able to integrate and manage heterogeneous data

from multiple sources guaranteeing maximal interoperability, and, being based on a microservice architecture, it offers wide scalability to allow its deployment in different sized cities, from small municipalities to big metropolitan areas. Ingested data is modelled into semantic entities based on classes defined in the KM4City ontology. Semantical indexing is sided with multiple big data storages (i.e., NoSQL databases) in order to store the ingested information. Specific APIs have been developed to allow fast data retrieval via relational, semantic and geographical queries. Processing logic and analytical services are available to process data and perform complex computations exploiting simulations engines and artificial intelligence solutions. Interactive dashboards are included to present data to the users and allow them to inspect and study the urban environment.

The architecture of Snap4City is based on multiple functional layers that cooperate to provide a complete solution for urban data management and analysis. The platform aims to offer high-quality services across several development phases.

Acquisition, Indexing and Storage: Manages the semantic ingestion of heterogeneous data from IoT sensors, municipal databases, open data, web services, and other sources [8], ensuring interoperability through the adoption of established standards, such as FIWARE Smart Data Models and IoT communication protocols.
Processing: Implements advanced analytical capabilities to produce high-level information including, for example, reconstruction of traffic density information from scattered data [9], prediction of air quality [10], assessment of public and sharing transportation performance [11], thus enabling the possibility to carry out what-if analyses to evaluate the impact of changes within the urban environment.
Visualization and Distribution: Creation of interactive 2D and 3D web interfaces accessible via standard browsers [12], which allow local authorities and citizens to explore the city's Digital Twin visualizing real-time and historic data and simulate alternative scenarios with high precision [13].

2.2 Modelling Building into the Snap4City Platform

The Snap4City platform is built upon the KM4City ontology. Smart city ontologies [14] play a pivotal role in both allowing semantic enrichment and supporting data interoperability [15]. Differently from other state-of-the-art ontologies for the smart city domain [16–20], in which entities usually are modelled with an articulated and overcomplex hierarchy of classes leading to an explosion of instances when a huge amount of data must be ingested and limiting the possibility to accommodate different kinds of data without frequent reengineering, KM4City adopted a different approach to facilitate the deployment and exploitation of semantic representation in SCDT platform.

Focusing on building representation, the *km4c:Building* class has been derived from a general *km4c:HighLevelType* (HLT) class. HLT is a broad concept that can be used to describe different kinds of entities relevant in the smart city context. It can be seen as a sort of label to add semantic information to data of different kinds. In addition, to support interoperability, the *km4c:Building* has also been subclassed by *s4city:CityObject* class from the SAREF4CITY ontology [21] (extension for smart cities of the IoT ontology SAREF [22]) and made equivalent to *bot:Building* of the Building Topology Ontology [23]. In Fig. 1**,** a simplified representation of the KM4City ontology is provided.

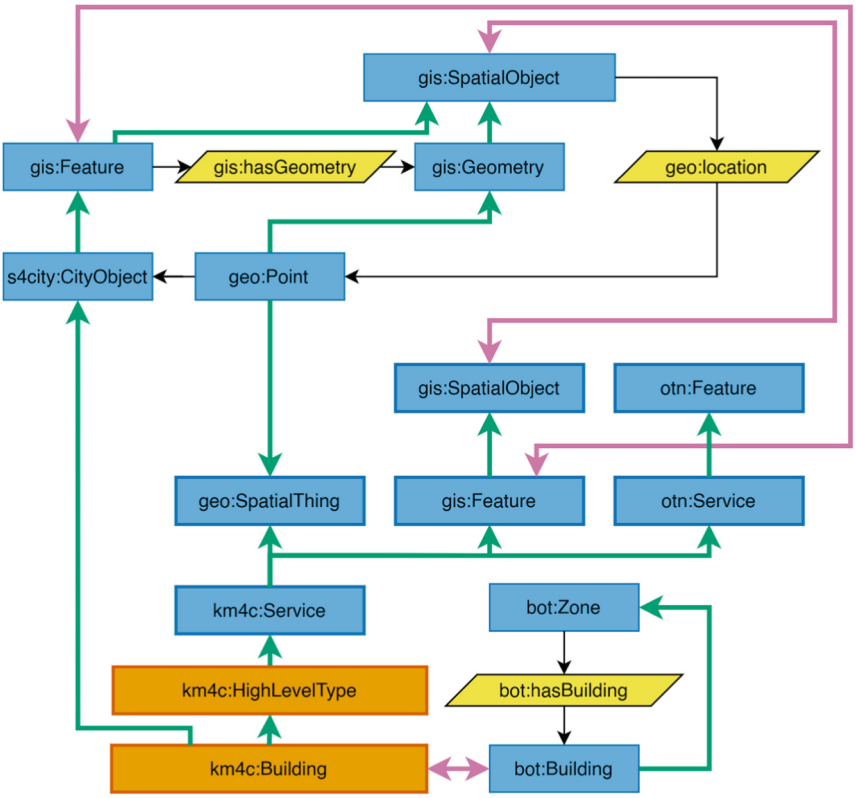


Fig. 1. Simplified diagram of the KM4City ontology focusing on *HighLevelType* and *Building* classes. Rectangles indicate classes, parallelograms are properties. Black arrows link triple relations, i.e., subject-property-object. Green arrows indicate superclass/subclass relations, while purple arrows show class equivalence. Prefixes are km4c for KM4City, bot for Building Topology Ontology, geo for WGS84, gis for GeoSPARQL, otn for Open Transport Network, s4city for SAREF4CITY.

Building entities are enriched with a set of additional data according to a defined data model composed by a set of static attributes and a number of variables each one semantically described with specific value type, value unit, and data types. For example, the address variable has value type *Geolocation*, value unit *text*, and data type *string*. In Table 1**,** the complete list of variables included in the data model of buildings, is reported.

Such data model allows to provide useful information for each specific building of the city, including its location (e.g., *address*, *city*, *postCode*), classification (*name*, *description*, *buildingType*), structural information (*numberOfFlats*, *numberOfLevel*, *numberOfUndergroundLevel*, *capacity*, *height*, *groundShape*, *isAccessible*), external sources (*osmID* from OpenStreetMap, *website*, *wikidata*, *wikipedia*), and available 3D models (*3DModel*, *3DModelType*, *3DModelDescription*). Note that multiple 3D models can be

Table 1. Variables of the building data model

Variable Name	Value Type	Value Unit	Data Type
dateObserved	Timestamp	ms	string
city	Geolocation	text	string
address	Geolocation	text	string
postCode	Identifier	ID	string
name	Identifier	ID	string
description	Status	text	string
buildingType	Status	coded status	string
numberOfFlats	Count	number	integer
numberOfLevel	Count	number	integer
numberOfUndergroundLevel	Count	number	integer
capacity	Count	number	integer
height	Status	number	float
groundShape	Geometry	complex data	JSON(GeoJSON)
isAccessible	Status	coded status	string
osmID	Identifier	ID	string
website	URL Link	webpage	string
wikidata	URI Link	ServiceURI	string
wikipedia	URL Link	webpage	string
3DModel	Identifier	ID	string
3DModelType	Status	coded status	string
3DModelDescription	Status	text	string

associated with the same building entity: for example, it is possible to have multiple representations at different Level of Detail (LoD), extending the classification of CityGML [24] by including also Building Information Models (BIM). In addition, associating multiple 3D representations enables the system to show temporal evolution of the same buildings, e.g., to monitor the progress of a construction site, or possible future changes yet to be built in the real environment, thus supporting urban development activities. Indeed, according to the data model, each building entity must include a *dateObserved* variable indicating the temporal instant in which the model refers to. In this way, it is possible to ingest the same building multiple times accumulating its evolution and possible future plans.

2.3 Massive Building Data Ingestion Pipeline

The main challenges in creating a urban digital twin capable to represent each single building as an independent entity are: (i) the data volume (a medium sized city like Florence, Italy, counts more than 50,000 buildings) requiring adequate platform architecture as well as fast and fault-tolerant ingestion pipelines able to recover from possible errors without requiring a full restart, (ii) the necessity to merge and reconciliate multiple data sources with robust mechanism to deal with missing data, and (iii) a visualization engine able to support massive rendering of 3D models. While characteristics of the rendering solution implemented into the Snap4City dashboards are described in Sect. 3, hereafter details on the ingestion pipeline are reported.

In the presented solution, two main sources of data are considered: OpenStreetMap (OSM), to recover building information and metadata, and a set of 3D building models reconstructed from Digital Terrains and Surface Models (DTM/DSM) following the method described in [25]. In particular, a local copy of OSM data have been used and instantiated into a Postgres database to allow faster data retrieval. Differently, 3D building models have been organized in a hierarchical structure mirroring the tiling system typically used in web maps. To allow data reconciliation each building model has been named with a prefix linked to the relative OSM ID.

Data ingestion has been implemented exploiting Snap4City Processing Logic/IoTApp (PLg) [26]. Snap4City PLg extends the Node-RED framework [27] with a series of specific functionalities (distributed as Node-RED libraries microservices) allowing full integration with platform services via APIs. Two different ingestion flows have been developed according to the complexity of the building: simple building, having a plant representable with a single polygon and indicated as *way* in OSM, and complex building, identified by the OSM *relations* and usually having plants composed by multiple polygons. Such a distinction is necessary to properly handle the different modalities in which OSM data are organized in the Postgres database for *ways* and *relations* and to manage the construction of the building plant geometry, eventually stored in the *groundShape* model variable (see Table 1) and into a knowledge base entity attribute in WKT format. As an example, the PLg flow for ingesting *way*-type buildings is shown in Fig. 2 and hereafter discussed.

The processing flow is composed of a sequence of steps (highlighted with gray boxes in Fig. 2). First, in step #0, variables to track the flow progress are initialized and available 3D models not having a match in the OSM database are identified and isolated. Eventually, such models will be used to create entities by providing required metadata manually.

After this first filter, in step #1, a batch of building IDs is selected and sent to the Queue Manager node. Such node keeps a queue of buildings to be processed and is responsible for releasing the next ID only after the ingestion process of the previous one is concluded (successfully or unsuccessfully). For each building ID its geographic coordinates are extracted from the OSM database in order to define its position and create its WKT geometry (used as shape representation in the knowledge base so as to enable geographic queries). Such data is used to define the building entity and, in step #2, is sent to the IoT Directory of the Snap4City platform to register the entity into the knowledge base. Once this operation successfully returns, in step #3 two operation are carried out.

First, actual building data are retrieved from the OSM database and inserted in the entity previously created, following the data model presented in Table 1**,** by sending the data to the internal context broker, thus initiating the ingestion process to store the information into a NoSQL database [8]. In parallel, the access policy of the entity is changed. Being a secure and privacy compliant platform, Snap4City enforces user ownership on any ingested data. By default, created entities are private and accessible only by the creator. Nevertheless, it is possible to make the entity public or delegate it in read, write, or read-and-write to other users. In the context of this work, all building entities have been made public.

To guarantee system robustness, the ingestion flow includes dedicated error-handling mechanisms with specific logging to register errors happening during the process. The managed error scenarios are three. The case of buildings without associated 3D models, shown in block E1 of Fig. 2; the failure of the IoT Directory during entity registration (illustrated in E2), and errors during data transmission to the context broker and subsequent data ingestion, highlighted in block E3. In the first case (E1) the flow logs the error and skips the specific building ingestion, since the 3D model is missing, it would not be possible to show it in the in 3D dashboard. In the other two cases recovery strategies have been devised. After a first failure, the operations are executed again for a given number of times after a short delay (set to 20 s) by re-running step #2 in case of error during entity creation, or step #3 in case of failed data ingestion. This is motivated by the fact that sometimes errors are related to temporary network failures, thus a simple retry could solve the problem. When such a strategy is not sufficient and the error persists, the failure is logged in order to be recovered in a successive time. Note that, in these cases, all the flow is halted for 10 min before moving on the next building. This solution has been designed to prevent the system from continuing sending requests in case of service problems in the IoT Directory to avoid overloading the platform infrastructure further.

Once all the batches have been executed, two different recovery strategies are executed to solve the logged errors. The first strategy, shown in block R1, is designed for recovering from errors that happened during entity creation: in this case steps #1, #2, and #3 all re-executed. Differently, errors during data ingestions are handled with block R2 that limits the re-execution to step #3. The recovery strategies start automatically for the first time. If failures happen again, these are logged by overwriting the error logs and the flow is stopped (exploiting a flow variable). This solution has been adopted since such repeated errors are probably due to some problems with the input data. Therefore, it is required that the operator correct the input and then manually restart the recovery flow and complete the ingestion. Note that both recovery strategies work on batches of IDs: while R1 injects the flow messages into step #1 exploiting its Queue Manager, R2 requires a different Queue Manager node, since steps #1 and #2 must not be executed in this case.

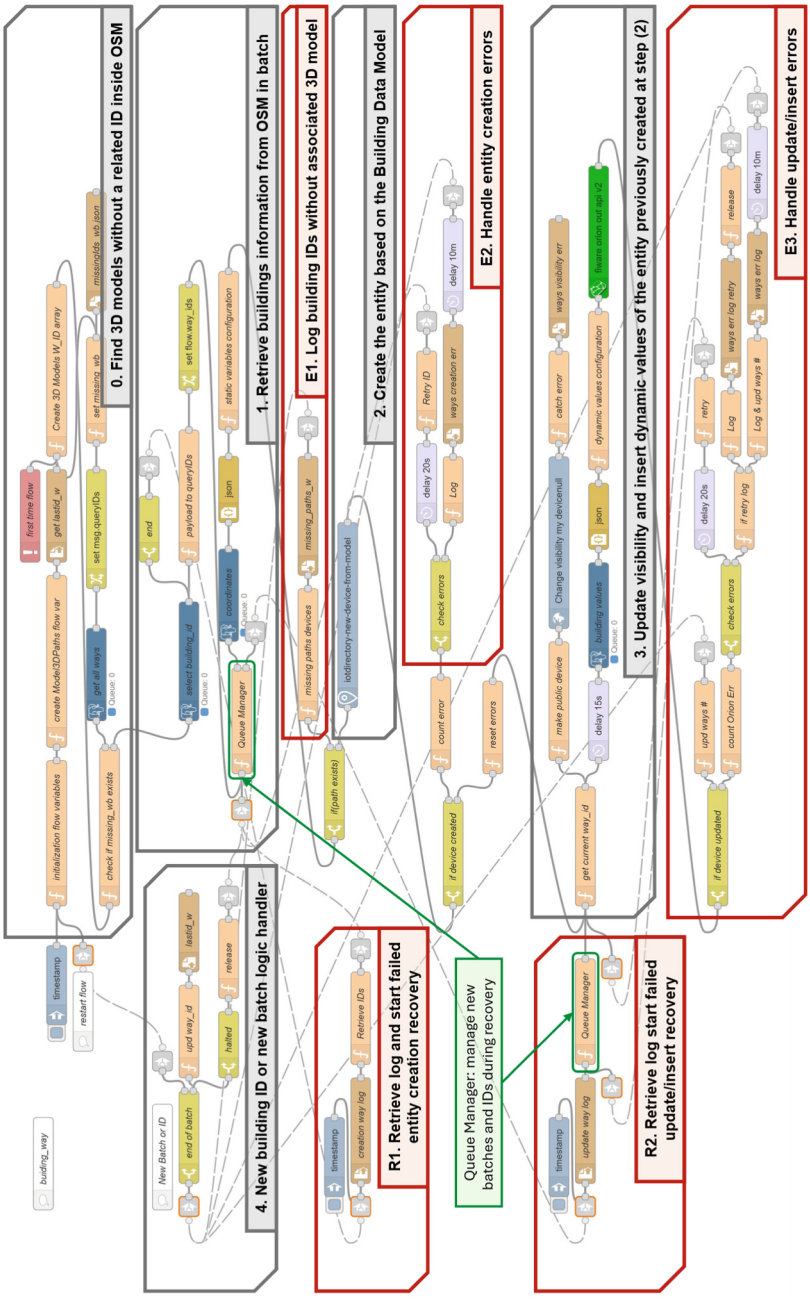


Fig. 2. Processing Logic ingestion flow for OSM ways. Different steps of the pipeline are highlighted with gray boxes and short descriptions. Differently sub-flows for error handling and recovery are highlighted in red.

3 Dynamic and Interactive 3D Visualization

The 3D representation of the urban Digital Twin is implemented through interactive dashboards in the Snap4City platform. In particular, the proposed solution exploits the Snap4City multi-data map widget, which is designed to integrate heterogeneous Smart City urban data in a 3D map visualization, such as: 3D building models at different LoD, terrain and elevation representation, orthomaps, road networks and traffic flows,

Points of Interest (POI), urban services, dynamic real-time heatmaps, environmental data and information from thousands of sensors distributed across the territory. All these features, together with the capabilities to perform predictive simulations and what-if analysis scenarios, transform the Digital Twin in the Snap4City environment into a fully operational tool, rather than a mere static representation of the city. An example of a Snap4City dashboard showing a multi-data map for 3D Digital Twin representation is shown in Fig. 3.

The graphic rendering in the Snap4City multi-data map widget is based on the Deck.gl framework, which provides the abstraction layer required to handle multiple categories of geographic data and 3D content in a web-based environment. As described in [2], this architecture has been extended with the introduction of elements, such as the *FusionLayer*, which are dedicated to improve the efficiency of data management and visualization for large urban scenarios. The *FusionLayer* is derived from the Deck.gl *TileLayer*, and it is responsible for handling tiled data in a smarter way, allowing for instance to reduce redundant server requests, and limiting client-side computational burden. This is achieved by exploiting client-side cache reuse through *topDownFusion* and *bottomUpFusion* methods that reuse already downloaded data across parent and child tiles during zoom-in and zoom-out operations, respectively. This is particularly relevant for 3D buildings, which require tiled representations and visualization at different zoom levels without providing excessive GPU load or undesirable data duplication. Accordingly, the proposed visualization framework preserves responsiveness even when thousands of buildings and multiple thematic layers are loaded simultaneously.

On top of the above-described architecture, several new interactive features have been introduced for the visualization and management of 3D building models. First, a dynamic loading mechanism has been implemented to replace the previous approach based on static JSON files. In the specific, whenever a request for showing 3D modeled buildings is made in the frontend, a backend script automatically retrieves the server directory structure associated with the currently visible tiles, in order to access the corresponding 3D model. This is done for each visible tile in the viewport. This mechanism ensures that the rendered 3D models are always updated and consistent with the server filesystem structure, since new models or modifications to existing ones become immediately available. This solution makes 3D model management and visualization more flexible and scalable, simplifying the integration of new models and the update of existing ones, without the necessity to manually update configuration files.

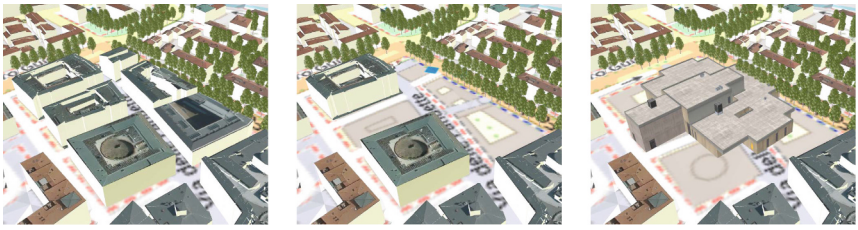
In addition, an interactive building selection and semantic data retrieval from the Knowledge Base have been added as new features: when the user clicks on a building in the 3D view, the selected object is highlighted, and the system queries the Knowledge Base to retrieve the semantic information associated with the selected building previously ingested according to the defined building data model (see Sect. 2 and Fig. 3(b)). In this



Fig. 3. Snap4City Dashboard showing a multi-data map for 3D Digital Twin representation. In (a) full view of the city center (rendering more than 30000 independent building entities) together with IoT sensors, traffic flow information, pollution heatmap. In (b) a close-up view on the Santa Maria del Fiore Cathedral that has been selected to show the related information in the right-sided panel.

way, an informative panel in the multi-data map can show both static and dynamic information related to the specific entity, such as address, building type, number of floors, height, capacity, and also external references (e.g., website, Wikipedia, Wikidata links). Therefore, the 3D model is not treated as a static graphic object, but as the visual counterpart of a semantically indexed urban entity.

Furthermore, the above-described informative panel is also designed to change the visualization model for the selected building at runtime: in addition to showing the metadata, the panel also lists the available 3D models (e.g., GLB/GLTF) for the specific building. When a different model is selected, the default model view is removed from the main scene graph and the new representation is instantiated in a dedicated layer for modified buildings, preserving the consistency between the main building layer and the modified one. As a consequence, the model substitution can be performed interactively without compromising the efficiency of the tiled rendering pipeline. Finally, it is possible to remove one or more buildings from the current scene, which is useful for implementing customized what-if scenarios altering the urban landscape. Actually, by temporarily hiding selected buildings, users can explore alternative urban configurations and visually assess the impact of urban design changes in the surrounding context. Moreover, by exploiting the combination of the model change and remove features, it is possible to introduce in the map new buildings spanning over an area previously covered by different buildings. This allows decision makers to evaluate the impact of renovating full neighborhood not limited to single constructions. An example of this feature is shown in Fig. 4.



(a) (b) (c)

Fig. 4. Example of building removal and full neighborhood modification. In (a) the original neighborhood. In (b) some on the building have been removed. In (c) all the original buildings have been removed leaving space for a new construction. All these changes can be executed directly on the dashboard by the user that select which 3D model visualize.

Therefore, all the above presented features allow the Snap4City dashboards to act as a front-end access point for inspecting semantically indexed entities and interacting with data-driven urban scenarios to monitor and evaluate possible changes of the urban environment.

4 Validation on the SCDT of Florence

To assess the effectiveness and scalability of the proposed framework, a comprehensive validation has been performed in the context of the realization of the SCDT of Florence, Italy. Florence is a medium-sized European city with a historically dense and morphologically complex urban environment, thus representing a challenging case study for urban digital twin applications.

The modelled area encompasses 51,029 buildings. Each building has been independently indexed as a semantic entity in the KM4City knowledge base, following the ingestion pipeline described in Sect. 2, and associated with multiple 3D models. Indeed, two buildings models (with high- and low-resolution roof textures) have been associated with all the buildings, and, when available, additional high quality 3D models and BIM have been considered for specific buildings. The buildings are spatially distributed across 786 tiles, collectively covering approximately 150 km^2, an extent wider than the administrative boundaries of the Florence municipality that includes surrounding areas. Using this scale allows to validate the framework in a concrete and real scenario, not limited to a proof-of-concept experiment.

The two ingestion flows (i.e., for ways and relations) have been executed exploiting a local copy of OSM and 3D models reconstructed from DTM/DSM data. The fault-tolerant design of the pipeline proved essential to efficiently ingest more than 50,000 entities. During ingestion, 169 OSM buildings have been found without an associated 3D model, thus discarded. On the other hand, a number of errors have been logged during the process: 468 errors happened due to temporary service unavailability and have been automatically recovered with the retry procedure. Differently, 84 errors required manual input corrections, after which the ingestion process concluded successfully.

The Snap4City multi-data map dashboard has demonstrated the ability to render the complete Florence digital twin interactively via web-browser. The dashboard is able to load and render thousands of independent building entities simultaneously together with additional data such as IoT sensors, traffic flow information, heatmap, etc. Ingested buildings have been correctly visualized and entity data is efficiently retrieved by clicking the 3D model, thus triggering API calls to perform knowledge base query and obtain entity attributes. Model substitution and removal have been tested thoroughly, evaluating also the possibility of altering entire urban blocks, thus confirming the effective operability of the what-if scenario mechanism.

These results support the conclusion that the proposed framework can be adopted in real scenarios to realize SCDT of medium and large municipalities.

5 Conclusions

In this paper a framework for the semantic ingestion and representation of 3D building entities within a Smart City Digital Twin has been presented. Exploiting the KM4City ontology and specific data models, the system is able to index buildings as semantically enriched knowledge base entities related to multiple 3D models, moving forward with respect to static 3D urban representations. In order to accomplish such a result, a scalable and fault-tolerant ingestion pipeline capable of reconciling OpenStreetMap metadata with 3D models has been developed using the processing logic built-in the Snap4City platform. In addition, dashboards within the Snap4City platform provide a high-performance 3D visualization environment that supports a wide range of dynamic interactions to allow users to select buildings and retrieve semantic data, substitute 3D models at runtime, remove structures, and redesign complete areas to simulate what-if scenarios. The framework has been validated on the case study of Florence, Italy, that required the management of over 50,000 independent building entities covering an

area of approximately 150 km^2. Therefore, the proposed solution, providing semantic representations of building and interactive widely accessible web-based 3D interfaces, realizes a comprehensive tool supporting decision-makers for the operational monitoring and strategic planning of smart cities, and allows common citizens to navigate through their city and be involved in the urban development.

Acknowledgments. This work has been carried out in the context of the Interreg Euro-MED TOURISMO project "TOURism Innovative and Sustainable Management of flOws", co-funded by the European Union (Euro-MED0200900).

Disclosure of Interests The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Taylor, J., Bennett, G., Mohammadi, N.: Engineering smarter cities with Smart City digital twins. J. Manag. Eng. **37**(6) (2021). https://doi.org/10.1061/(asce)me.1943-5479.0000974
2. Adreani, L., Bellini, P., Fanfani, M., Nesi, P., Pantaleo, G.: Smart City digital twin framework for real-time multi-data integration and wide public distribution. IEEE Access. **12**, 76277–76303 (2024). https://doi.org/10.1109/ACCESS.2024.3406795
3. Nesi, P., Garau, C., Zamperlin, P.: Ottimizzazione delle Città Intelligenti: L'Utilizzo Dei gemelli Digitali e dell'Intelligenza Artificiale nei Processi Decisionali Urbani. CONTESTI. **2**, 1–13 (2025)
4. Cheng, R., Hou, L., Xu, S.: A review of digital twin applications in civil and infrastructure emergency management. Buildings. **13**(5), 1143 (2023). https://doi.org/10.3390/buildings13051143
5. Cureton, P., Hartley, E.: City information models (CIMs) as precursors for urban digital twins (UDTs): a case study of Lancaster. Front. Built Environ. **9** (2023). https://doi.org/10.3389/fbuil.2023.1048510
6. KM4City, the DISIT knowledge model for city and mobility. Available at http://wlode.disit.org/WLODE/extract?url=http://www.disit.org/km4city/schema. Accessed 3 Mar 2026
7. Bellini, P., Benigni, M., Billero, R., Nesi, P., Rauch, N.: Km4City ontology building vs data harvesting and cleaning for smart-city services. J. Vis. Lang. Comput. **25**(6), 827–839 (2014). https://doi.org/10.1016/j.jvlc.2014.10.023
8. Bellini, P., Collini, E., Fanfani, M., Nesi, P., Panconi, C.: Efficient and scalable semantic data ingestion for Smart City digital twin platforms. In: 2025 IEEE 11th International Conference on Big Data Computing Service and Machine Learning Applications (BigDataService), Tucson, AZ, USA, pp. 44–51 (2025). https://doi.org/10.1109/BigDataService65758.2025.00012
9. Bilotta, S., Nesi, P.: Traffic flow reconstruction by solving indeterminacy on traffic distribution at junctions. Futur. Gener. Comput. Syst. **114**, 649–660., ISSN 0167-739X, (2021). https://doi.org/10.1016/j.future.2020.08.017
10. Bilotta, S., Ipsaro Palesi, L.A., Nesi, P.: Exploiting open data for CO2 estimation via artificial intelligence and eXplainable AI, 128598., ISSN 0957-4174. Expert Syst. Appl. **291** (2025). https://doi.org/10.1016/j.eswa.2025.128598
11. Fanfani, M., Ipsaro Palesi, L.A., Nesi, P.: Data-driven mobility demand-versus-offer matching by dynamic integration of public and sharing services. IEEE Access. **14**, 23166–23193 (2026). https://doi.org/10.1109/ACCESS.2026.3661850

12. Bellini, P., Fanfani, M., Nesi, P., Pantaleo, G.: Snap4City dashboard manager: a tool for creating and distributing complex and interactive dashboards with no or low coding, 101729., ISSN 2352-7110. SoftwareX. **26** (2024). https://doi.org/10.1016/j.softx.2024.101729
13. Adreani, L. et al.: Smart City scenario editor for general what-if analysis. Sensors. **24**, 2225 (2024). https://doi.org/10.3390/s24072225
14. De Nicola, A., Villani, M.L.: Smart City ontologies and their applications: a systematic literature review. Sustainability. **10**(10), 5578 (2021). https://doi.org/10.3390/su13105578
15. Bittner, T., Donnelly, M., Winter, S.: Ontology and semantic interoperability. In: Large-Scale 3D Data Integration. CRC Press (2005)
16. Fernandez, S., Hadfi, R., Ito, T., Marsa-Maestre, I., Velasco, J.R.: Ontology-based architecture for intelligent transportation systems using a traffic sensor network. Sensors. **16**(8), 1287 (2016)
17. Shi, J., Pan, Z., Jiang, L., Zhai, X.: An ontology-based methodology to establish city information model of digital twin city by merging BIM, GIS and IoT. Adv. Eng. Inform. **57**, 102114 (2023). https://doi.org/10.1016/j.aei.2023.102114
18. Ju, J., Liu, L., Feng, Y.: Citizen-centered big data analysis-driven governance intelligence framework for smart cities. Telecommun. Policy. **42**(10), 881–896 (2018). https://doi.org/10.1016/j.telpol.2018.01.003
19. D'Aniello, G., Gaeta, M., Orciuoli, F., Sansonetti, G., Sorgente, F.: Knowledge-based Smart City service system. Electronics. **9**(6), 965 (2020). https://doi.org/10.3390/electronics9060965
20. Sobral, T., Galvão, T., Borges, J.: Knowledge-assisted visualization of multi-level origin-destination flows using ontologies. IEEE Trans. Intell. Transp. Syst. **22**(4), 2168–2177 (2021). https://doi.org/10.1109/TITS.2021.3056228
21. s4city: SAREF extention for smart city. Available at https://saref.etsi.org/saref4city/v1.1.2/. Accessed 3 Mar 2026
22. SAREF: The smart applications REFerence onotology. Available at https://saref.etsi.org/core/v3.2.1/. Accessed 3 Mar 2026
23. Building Topology Ontology. Available at https://w3c-lbd-cg.github.io/bot/. Accessed 3 Mar 2026
24. Gröger, G., Plümer, L.: CityGML–interoperable semantic 3D city models. ISPRS J. Photogramm. Remote Sens. **71**, 12–33 (2012). https://doi.org/10.1016/j.isprsjprs.2012.04.004
25. Adreani, L., Bellini, P., Colombo, C., et al.: Implementing integrated digital twin modelling and representation into the Snap4City platform for smart city solutions. Multimed. Tools Appl. **83**, 37121–37146 (2024). https://doi.org/10.1007/s11042-023-16838-0
26. Fanfani, M., Ipsaro Palesi, L.A., Nesi, P.: Microservices' libraries enabling server-side business logic visual programming for digital twins, 101805., ISSN 2352-7110. SoftwareX. **27** (2024). https://doi.org/10.1016/j.softx.2024.101805
27. Node RED framework. Available at https://nodered.org/. Accessed 3 Mar 2026