Storing and Visualising Dynamic Data in the Context of Energy Analysis in the Smart Cities

16

Thunyathep Santhanavanich, Rosanny Sihombing, Pithon Macharia Kabiro, Patrick Würstle, and Sabo Kwado Sini

Abstract

There is increased activity in developing workflows and implementations in the context of urban energy analysis simulation based on 3D city models in smart cities. At the University of Applied Sciences Stuttgart (HFT Stuttgart), an urban energy simulation platform called 'SimStadt' has successfully been developed. It uses the CityGML 3D city model to simulate the heat demand, photovoltaic potential, and other scenarios that provide dynamic simulation results in both space and time dimensions. Accordingly, a tool for managing dynamic data of the CityGML models is required. Earlier, the CityGML Application Domain Extension (ADE) had been proposed to support additional attributes of the CityGML model; however, there is still a lack of open-source tools and platforms to manage and distribute the CityGML ADE data efficiently. This article evaluates and compares alternative methods to manage dynamic simulation results of the 3D city model and visualise these data on the 3D web-based smart city application, including the use of SimStadt web services, databases, and OGC SensorThings API standard.

Keywords

Urban energy analysis · 3D building model · CityGML · 3D tiles · SensorThings · Smart cities · SimStadt

T. Santhanavanich (✉) · R. Sihombing · P. M. Kabiro · P. Würstle · S. K. Sini
Center for Geodesy and Geoinformatics, Hochschule für Technik Stuttgart, Stuttgart, Germany
e-mail: thunyathep.santhanavanich@hft-stuttgart.de

V. Coors et al. (eds.), *iCity. Transformative Research for the Livable, Intelligent, and Sustainable City*,
https://doi.org/10.1007/978-3-030-92096-8_16

Abbreviation

3DCityDB	3D City Databases
HFT Stuttgart	Hochschule für Technik Stuttgart (de) or University of Applied Sciences Stuttgart (en)
HTML	HyperText Markup Language
INSPIRE	Infrastructure for Spatial Information in the European Community
NPM	Node Package Manager
OGC	Open Geospatial Consortium
PV	Photovoltaic
REST	Representational state transfer
STA	SensorThings API
UML	Unified Modeling Language
XML	Extensible Markup Language

16.1 Introduction

As cities continue the implementation of smart city concepts around the world, a smart city has been defined as a way of continuously optimising traditional services and networks by taking advantage of developments in Information and Communications Technology (ICT) to become more efficient to benefit its inhabitants (European Commission 2020). Smart cities are often associated with the intelligent network of connected objects and machines that continuously transmit data using sensors technology and the cloud (Jawhar et al. 2018). At the same time, the virtual 3D city model has been used predominantly in the past for visualisation (Biljecki et al. 2015). There is a need to visualise and analyse city data alongside the 3D virtual city model since the 3D city model could also contain essential datasets related to individual buildings that can altogether be used in helping municipalities, enterprises, and citizens make better decisions that improve the quality of life in their cities.

While different standards are used to model and manage 3D city models, having a common standard eases the exchange of this data between various partners, thereby making these data reusable. To have a standard definition of the basic entities, attributes, and relationships of a 3D city model, the Open Geospatial Consortium (OGC) CityGML was developed to enable not just the visualisation of the virtual 3D city model but also the management and sharing of these models (Gröger et al. 2012). As one of the popular standards, CityGML has been widely accepted and used for modelling and sharing the 3D city model (Arroyo Ohori et al. 2018).

With CityGML models used to store building energy-related information like building function and year of construction which are vital for building energy simulation, various energy simulation platforms like SimStadt (Schumacher 2020) have adopted CityGML as a

definitive source for 3D city model information as data input for performing energy simulations such as energy demand and solar energy potential. These simulation results are usually represented by the visualisation platform as it makes data more comfortable for the human to understand and makes it easier to detect patterns, trends, and outliers in groups of data (Yi et al. 2008). Furthermore, city administrators and agencies can, therefore, use these data visualisations to make important decisions concerning their cities and make changes where needed to improve efficiency.

However, the various data simulation results from CityGML-based building models could lead to the complexity and heterogeneity of the data model. Therefore, a proper way of managing and visualising this information is necessary and needed to trace patterns and place meaning within the data being integrated. In the past, the Application Domain Extension (ADE) had been developed for extending and managing a specific group of environmental data such as energy-related building data to be modelled in connection to CityGML (Biljecki et al. 2018). Still, ADE has some limitations as it needs layers of data conversion, structures, and tools to access, deliver, and visualise on the web client (Lim et al. 2020).

In this article, we study and evaluate alternative approaches for managing energy-related building information with a particular focus on how the 3D city model can be used for collecting, computing, and visualising energy simulations using the following methods: (1) computing and visualising the simulated energy data of 3D building models on-the-fly, (2) using the PostgreSQL database as a datastore for simulated energy data, and (3) using SensorThings for managing the simulated energy data. These methods are, however, not the only methods available, but we intend to compare them to find out which is more efficient when it comes to managing and visualising energy-related building information such as photovoltaic (PV) energy generation potential, heat demand, etc. from CityGML models.

16.2 Background

16.2.1 Energy Data Simulation of the 3D Building Models

SimStadt is a simulation software developed at HFT Stuttgart. This software is based on the modules from the INSEL block diagram simulation system (Schumacher 2020). It is used to create workflows to simulate the dynamic energy-related attributes in the 3D city models in CityGML format (Monsalvete et al. 2015). CityGML is an OGC standard format to store and exchange city models based on the Extensible Markup Language (XML) format, which contains 3D urban geometry description and other metadata.

An example of a workflow from the SimStadt simulation platform is the heating demand workflow, which is based on the monthly energy balance. This workflow requires three building parameters extracted from the input CityGML data: geometric data, building physics attributes, and building usage attributes. To calculate the building physics

attributes, information from the CityGML attributes *yearOfConstruction* and *function* are required. These two sets of information are then used to categorise buildings based on their type and age (Nouvel et al. 2015). Information about the age of a building is used in calculating the thermal transmittance of walls, roofs, floors, and ceiling surfaces (Agugiaro 2016; Zirak et al. 2020).

16.2.2 Energy Data Management

CityGML Application Domain Extension

The possibility for CityGML to be extended through the ADE mechanism enables other information to be modelled along with the already existing real-world 3D model. Since the availability of this possibility, several pieces of information have been modelled, which includes Energy ADE and Utility Network ADE (Kolbe et al. 2011). Energy ADE extends the CityGML standard by features and properties, which are necessary to perform energy simulation and for storing the corresponding results (Gröger et al. 2012). With the objectives of managing and storing data required for calculating building energy simulation and results, Energy ADE provides a holistic approach for managing energy-related information. It can also be used not just for a detailed single-building energy simulation but also for city-wide, bottom-up energy assessments, focusing specifically on the buildings sector (Agugiaro et al. 2018). There are several applications for exploring a CityGML dataset, but most use cases have to convert the CityGML to another format such as glTF, 3D Tiles, or i3s for web visualisation. After the conversion process, the data and information linked to CityGML ADE are lost. For example, the 3D City Database (3DCityDB) has implemented ADE support into its database. However, users have to develop their mapping script for reading the ADE contents from the database and matching these datasets to the viewer format (Yao et al. 2018).

In 2015, research on storing and exchanging sensor or time-series data in the CityGML model had been conducted with a concept referred to as 'Dynamizer' as one of the CityGML ADE (Chaturvedi et al. 2015). It is used to model and implement the dynamic properties for semantic 3D city models. It allows representing dynamic and time-varying attributes directly in the 3D city model in CityGML format. It supports encodings of the dynamic data by Domain-Range encoding and by Time-Value pair encoding in XML format. Each CityGML model can contain several representation encodings of the dynamic data. Example use cases of using Dynamizers had been implemented to connect the dynamic properties of the building, such as heat demand or energy generation from the attached solar panels. However, there are still limitations to the use of the Dynamizers concept. For example, the data manager must have access to the 3D city model, which may prove difficult when the city model data and simulated data are managed by different parties or organisations. Also, tools for parsing supporting encoding types of the Dynamizers written in XML are needed to access the dynamic contents. The data conversion and data structures still have to be implemented in order to utilise the data efficiently.

SensorThings API (STA)

The SensorThings API is one of the OGC standards of a protocol that unifies ways to interconnect the Internet of Things (IoT) devices, data, and applications over the web. SensorThings has two main parts which are Sensing and Tasking. The Sensing part provides an easy-to-use representational state transfer (REST) application programming interface (API) for managing the heterogeneous data. These operations include HTTP *POST, GET, PATCH,* and *DELETE* to create, read, update, and delete the sensor data and metadata, respectively (Liang et al. 2016). Recently, SensorThings API has been used by several domains. For example, SensorThings has been used as a service for managing heterogenous air quality sensor data in the European Union Infrastructure for Spatial Information in the European Community (INSPIRE) (Kotsev et al. 2018) and managing COVID-19 statistics (Santhanavanich et al. 2020). Additionally, the SensorThings API is expandable to manage dynamic time-series datasets in the CityGML 3D city models; the systematic study on this topic was conducted by Santhanavanich and Coors (2021).

16.2.3 3D Data Visualisation (Digital Globe)

The development of web applications for visualisation of 3D objects is built upon the foundation of the web technologies HTML5 and WebGL. HTML5 introduced the *canvas* element, which, when coupled with JavaScript, allows graphics to be drawn by web browsers while taking advantage of the multi-threading capability of modern browsers. WebGL extends the *canvas* element and allows for the rendering of 3D graphics without the need for plugins and extensions (Chaturvedi et al. 2015). The Cesium JavaScript library has been developed with this vision of the 'digital earth' in mind; it supports 3D data natively, it is able to portray massive amounts of data, and it allows users to combine heterogeneous datasets (Moore, 2018). From a technical perspective, Cesium may be described as an imperative high-level JavaScript library built on top of WebGL that provides a mapping API that is considered a suitable replacement for the now deprecated Google Earth API (Hoetmer 2014; Krämer and Gutbell 2015). In recent research, Würstle et al. (2020) have shown a proof of concept for visualising the 3D city models with the simulated energy data from SimStadt in the CesiumJS WebGL framework.

16.3 Concept

This section explains our concept for managing the simulated energy data of 3D building models using the SimStadt simulation software. Several approaches had been implemented and these are compared and evaluated in turn. These approaches include (1) computing and visualising the simulated data on the fly, (2) using the PostgreSQL database as a datastore

for the simulated energy data of 3D building models, and (3) using SensorThings for managing the simulated energy data of 3D building models.

16.3.1 Computing and Visualising the Simulated Energy Data of 3D Building Models on-the-Fly

This approach uses SimStadt web service to run energy simulations using only a web browser and network connection. Therefore, this approach makes the energy analysis available through a network regardless of the operating system of the running devices. The end users on this approach define the input parameter values on an HTML (HyperText Markup Language) form and then submit them to SimStadt to run the energy analysis. SimStadt will receive these values as a request to run an energy analysis. If the process is successful, SimStadt will respond to the energy analysis request with the analysis result data in JSON format. Afterwards, the analysis result data must be extracted and mapped to fit the 3D Tiles colouring scheme so that each building in the 3D building model can be coloured based on its corresponding analysis result for the geovisualisation purpose. Mapping the result data to the 3D Tiles should be done as many times as the number of data categories for the 3D geovisualisation. In this approach, there is no mechanism to store the result data once the users stop the web-based application by closing the web browser. Therefore, in a new web session, the whole process must be repeated when a user runs an energy simulation process, even though the result for the selected building or area might be the same as the previous energy simulation request.

16.3.2 Using the PostgreSQL Database as a Datastore for the Simulated Energy Data of 3D Building Models

The main aspect of this approach is the use of the database for managing the energy data and building relevant information. The 3DCityDB has an implementation for PostgreSQL, which in this approach, is used to store the 3D city models in the CityGML format. The database functions as an anchor point for the SimStadt simulation software. The SimStadt platform requires building information from the CityGML model to run its workflows. This approach allows us to keep everything up to date in a centralised way where not every file needs to be updated if it is used solely for visualisation purposes. The connection between the database and the simulation software is bidirectional. The database provides the input data to the simulation software, and the software stores its output in the database. Through the visualisation, users can update values in the database for the simulation.

16.3.3 Using SensorThings for Managing the Simulated Energy Data of 3D Building Models

This approach is based on the CityThings from our previous work (Santhanavanich and Coors 2021), which uses the SensorThings as a standardised specification API for managing the dynamic energy data in the 3D city models. The concept of this approach is to precalculate the energy-related data of the 3D city models with SimStadt. Then, the JSON result from SimStadt is constructed to conform with the SensorThings and its entities' specification. The produced JSON result is then added or updated to the SensorThings server with the HTTP *POST* or *PATCH* operations, respectively. On the client-side, users can directly request the energy data of each 3D building in JSON format from the SensorThings server with the HTTP *GET* request. This result of energy data is used to map to the 3D city models for the data visualisation.

16.4 Implementation

The implementations of this research are based on the 3D city models in the CityGML format. First, the building simulations were performed in the SimStadt simulation platform based on these CityGML building models (see Sect. 16.4.1). Second, three different approaches for managing the energy-related result from SimStadt and distributing to the web clients were conducted (see Sect. 16.4.2). Finally, the 3D building models with energy data were visualised on the web-based clients for each data management approach. The implementations of each approach in this paper are described in Fig. 16.1.

After servers had been implemented according to the three mentioned approaches, a 3D web application had been developed to present the result data by colouring the building roofs using the data and also showing them in an informative table (see Fig. 16.2). This table is displayed when users select a particular roof in the application. The building roofs are symbolised in different colours based on six different result data, where users can switch between to have a more intuitive result, which should help users to interpret the result data. These result data are PV potential yield, PV specific yield, levelised cost of electricity, total investment, discounted payback period, and financial feasibility.

16.4.1 Energy Simulation of the 3D Building Models with SimStadt Software

This study uses SimStadt to simulate the buildings' PV potential and heat demand analysis to demonstrate the management of the dynamic data from a building energy analysis. The 3D building model is the basis of the calculation in SimStadt. SimStadt extracts the geometric and semantic data from the CityGML of the observed area for the simulation process. Furthermore, it also requires the local weather data of the observed area on an

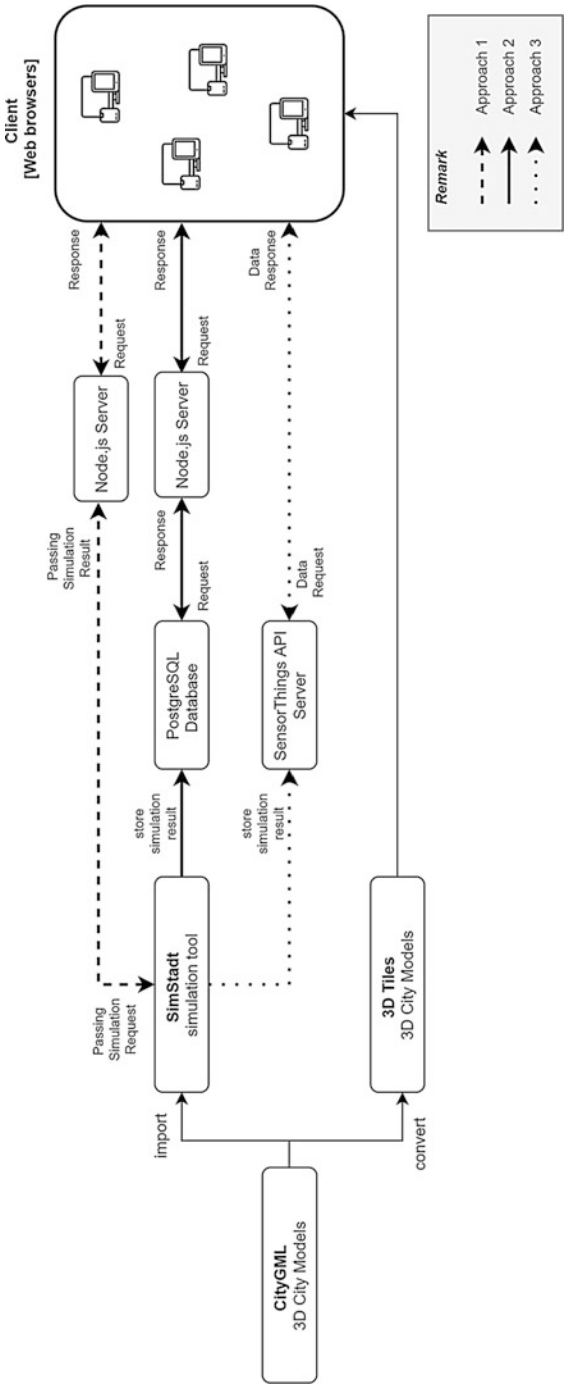

Fig. 16.1 Overall system architecture

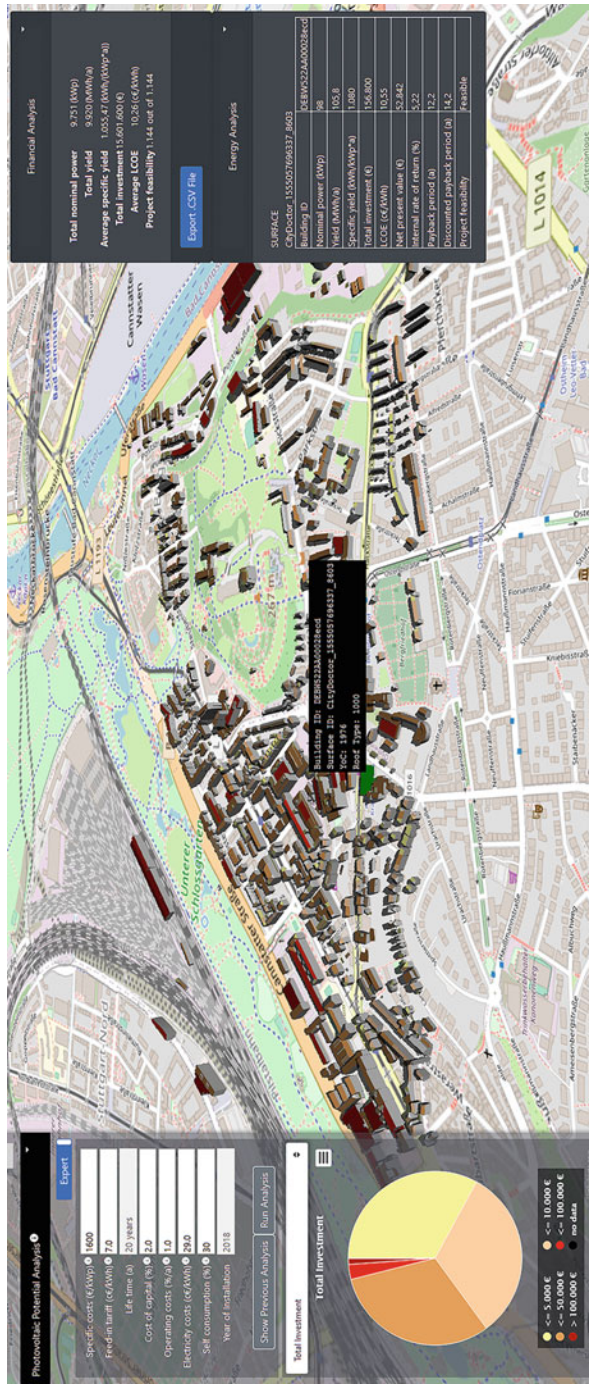

Fig. 16.2 A 3D web application for visualising the 3D building models in the area of Stöckach, Stuttgart with the simulated total investment cost to install PV system using SimStadt simulation API

hourly and monthly basis, including ambient temperature, relative humidity, horizontal global, and diffuse radiations, which can be imported from a third-party data provider (e.g. PVGIS, Insel) or weather data files (Meteonorm). In the on-the-fly approach (see Sect. 16.3.1) and the SensorThings approach (see Sect. 16.3.3), we simulated buildings' PV potential and the financial feasibility of the photovoltaic system in Stöckach, a city quarter of Stuttgart, Germany. We implemented the approach with database (see Sect. 16.3.2) by simulating buildings' heat demand of Ludwigsburg, Germany.

16.4.2 Managing Simulated Energy Data of 3D Building Models

Approach 1: Managing Simulated Energy Data of 3D Building Models on-the-Fly

In this approach, users are asked to define the mandatory input parameters on the 3D web platform. Afterwards, the system translates the user-defined input values into a web service request to SimStadt to perform a PV potential and feasibility simulation based on the user-defined input values. Upon a successful simulation process, SimStadt sends back the simulation result data in JSON format. This result consists of 125 roof surfaces, and each surface has 38 name/value pairs consisting the roof details, such as building ID, surface ID, coordinate, azimuth, tilt, and PV potential and feasibility analysis-related details, such as PV potential yield, irradiance on roof plane from January until December, total investment, maintenance costs, and financial feasibility.

Approach 2: Managing Simulated Energy Data of 3D Building Models Using a Database

This approach uses three main components. The main part of this approach is the database which is a PostgreSQL implementation as illustrated in Fig. 16.1. The 3DCityDB can import CityGML building models to the database using a 3DCityDB Importer tool. A tool was created to add the energy-related data to the database. The supported SimStadt workflows for this approach are solar potential and energy demand. The energy demand workflow consists of six individual modules. To connect from SimStadt to the database, the workflow needs to be extended. The third module in this workflow is a preprocessing step where the model is prepared for the further calculations which are done in the following modules. The specifically programmed extension is used after the last preprocessing step and before the first calculation step.

Approach 3: Managing Simulated Energy Data of 3D Building Models Using OGC SensorThings API

This approach uses SensorThings as a standard interface to manage and distribute the simulated energy data of 3D building models. This includes the installation of the SensorThings on the server following by the data modelling based on the entities model of the SensorThings standard specification. Then, the data and metadata of the simulation

tool and simulation result are stored to the SensorThings with the SensorThings Manager tool (STA Manager). Finally, the testing is conducted before sharing and connecting the SensorThings to the clients.

The simulation result from the SensorThings can be requested by any client through the HTTP *GET* request according to the SensorThings standard specification in the standard document (Liang et al. 2016) with the CityThings concept (Santhanavanich and Coors 2019). For the visualisation, the 3D city models in 3D Tiles format were loaded in the CesiumJS-based application. These 3D Tiles models preserved the *gml_id* attributes as identification of each building. Accordingly, clients can request for all dynamic contents of each building by specifying the *gml_id* of the CityGML building model.

16.5 Evaluation

This research evaluates and compares three different approaches for managing the energy-related data of the 3D city models including (1) managing building simulation data on-the-fly (via simulation API), (2) managing building simulation data using a database, and (3) managing building simulation data using OGC SensorThings API standard. Several features in aspects of data visualisation on the web client and data organisation had been conducted (see Table 16.1).

In the first approach, the simulation is done on-the-fly in real time. The main advantage of this approach over other approaches is that users can input the parameters for the building energy simulation every time they use the application. However, as the simulated data is not stored anywhere, the energy simulation process is always triggered every time users visualise the energy data, which costs the server computation and user payload.

In the second approach, the regular database is used for storing and distributing the energy-related data of the building models. This provides an advantage when storing large-scale building data. Also, the simulation data can be loaded to the client in a very short time as the simulation data is pre-calculated on the server-side. However, as there is no interface to get the data from the database, it needs a programming tool to connect a client to the database level, which this process has to be repeated when applying to a new use case. Another obvious drawback is that users cannot input the parameters to calculate specified simulation scenarios on-the-fly. To deal with this, several simulation scenarios can be pre-calculated with a trade-off for the storage requirement on the database.

In the last approach, the SensorThings API specification is implemented on the server-side for managing the energy-related data of the building models in which each *Thing* entity is linked to the particular CityGML part by the CityThings concept (Santhanavanich and Coors 2021). As the backend of the SensorThings server is the database, it supports large-scale building energy data transaction. In this approach, users can request, add, update, or delete energy data of the building models from the database through the SensorThings interface, which is a standardised specification from OGC (Liang et al. 2016). Accordingly, the data managed by SensorThings can be distributed widely for

Table 16.1 The overview feature comparison of three different approaches for organising the building simulation data

Data management Features		Managing simulated energy data of 3D building models on-the-fly	Managing simulated energy data of 3D building models using a database	Managing simulated energy data of 3D building models using OGC SensorThings API
Data visualisation visualization	User interface for visualising building simulation on the web client	Users can input any parameters or scenarios on-the-fly	Users can only select pre-simulated scenarios	Users can only select pre-simulated scenarios
	User loading time for visualising building simulation	~20–30 s per 100 buildings	~1 to 2 s per 100 buildings	~1 to 2 s per 100 buildings
Data organisation	Support data storage in the database	–	Supported	Supported
	Management of the multiple building simulation scenarios in the database	–	Manually manage with an additional application layer, e.g. node.Js server	Organised by SensorThings data model in the *Datastream* entity
	Role-based data access to the simulation data	–	Accessible only to the database administrators. Role-based access control can be extended	Different authentication roles can be given to users (read, write, update, delete, and admin)
	Interface to access, add, update, and delete the data in the database	–	Manually manage with an additional application layer, e.g. Node.js server	Users with appropriate roles can access, add, update, and delete data through the REST-based protocol
	Building simulation data distribution	–	The API to allow data access can be extended. The API document must be added	The data can be access through SensorThings interface. The standard API document is provided by OGC

several client applications with the SensorThings interface without the need for the dynamic web server to access the database. Although it lacks flexibility for user's specified parameters, several simulation scenarios can be performed and updated to the SensorThings server effectively through the HTTP *POST* method. Compared to the CityGML ADE, SensorThings manages the energy data independently, and there is no need for modifying the existing CityGML XML schema for the energy data. Moreover, the SensorThings has the JSON-based encoding, and its interface supports queries through the HTTP *GET* request, which includes result sorting, result filtering, and geolocation filtering. Still, the data modelling of the energy data to the SensorThings schema is not straightforward, since the characteristic of dynamic data from an energy simulation is different from the ones from IoT devices. Therefore, the mapping must be carefully designed in order to reach optimal usage.

16.6 Conclusion

CityGML models are used in several domains and locations around the world. In the context of the building energy demand and PV potential simulation, the CityGML ADE had been developed to handle the simulation result. However, there is still a lack of open-source tools and platforms to manage and distribute data efficiently. In this article, we present a way to bridge these gaps by evaluating and comparing three alternative approaches for managing the building energy simulation data, including (1) managing simulation data on-the-fly (via simulation API), (2) managing simulation data using a database, and (3) managing simulation data using OGC SensorThings API standard. The three implementations were assessed comparatively in the use cases of heat demand and PV potential analysis in cities in Germany, including Stöckach-Stuttgart, Ludwigsburg, and Grünbühl. The evaluation of these three approaches is discussed in Chap. 5. In summary, each approach discussed in this research has advantages and disadvantages that can effectively influence which approach will be selected by researchers or application developers.

The first approach would give users the flexibility to simulate the building energy demand and potential with user-specified parameters in which the high-performance computation power is needed on the server-side. However, in the second and third approaches, the database is used to store the simulated result in advance, in which high disk space on the server is required to store several simulation scenarios. Comparing the regular database and the SensorThings for storing the simulated result, the SensorThings shows advantages of flexibility, allowing users with multiple roles to read, add, update, and delete the data with the standardised request protocol. Accordingly, using SensorThings is more efficient and effective in terms of data management and data distribution. In conclusion, the integration of the SimStadt API service for simulating results on-the-fly and the SensorThings API for managing pre-simulated results is recommended to enable most benefits to store and visualise the energy analysed data in the smart cities' application.

References

Agugiaro, G., Benner, J., Cipriano, P., & Nouvel, R. (2018). The Energy Application Domain Extension for CityGML: enhancing interoperability for urban energy simulations. Open Geospatial Data, Software and Standards, 3(1). https://doi.org/10.1186/s40965-018-0042-y

Arroyo Ohori, K., Biljecki, F., Kumar, K., Ledoux, H., & Stoter, J. (2018). Modeling Cities and Landscapes in 3D with CityGML. In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), Building Information Modeling (pp. 199–215). Springer International Publishing. https://doi.org/10.1007/978-3-319-92862-3_11.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., & Çöltekin, A. (2015). Applications of 3D City Models: State of the Art Review. ISPRS International Journal of Geo-Information, 4(4), 2842–2889. https://doi.org/10.3390/ijgi4042842.

Biljecki, F., Kumar, K., & Nagel, C. (2018). CityGML Application Domain Extension (ADE): overview of developments. *Open Geospatial Data, Software and Standards*, 3(1). https://doi.org/10.1186/s40965-018-0055-6.

Agugiaro, G. (2016). Energy planning tools and CityGML-based 3D virtual city models: experiences from Trento (Italy). *Applied Geomatics*, 8(1), 41-56. https://doi.org/10.1007/s12518-015-0163-2.

Chaturvedi, K., Yao, Z., & Kolbe, T. H. (2015). Web-based Exploration of and interaction with large and deeply structured semantic 3D city models using HTML5 and WebGL. In *Bridging Scales-Skalenübergreifende Nah-und Fernerkundungsmethoden*, 35. Wissenschaftlich-Technische Jahrestagung der DGPF.

European Commission. (2020). Smart cities: Cities using technological solutions to improve the management and efficiency of the urban environment. https://ec.europa.eu/info/eu-regional-and-urban-development/topics/cities-and-urban-development/city-initiatives/smart-cities_en. Accessed 14 September 2020.

Gröger, G., Kolbe, T.H., Nagel, C. & Häfele, K.-H. (2012). OGC City Geography Markup Language (CityGML) Encoding Standard. Version:2.0.0. OGC 12-019. http://www.opengis.net/spec/citygml/2.0. Accessed 2 Nov. 2018.

Hoetmer, K. (2014) *Announcing deprecation of the Google Earth API.* https://mapsplatform.googleblog.com/2014/12/announcing-deprecation-of-google-earth.html. Accessed 14 September 2020.

Jawhar, I., Mohamed, N., & Al-Jaroodi, J. (2018). Networking architectures and protocols for smart city systems. Journal of Internet Services and Applications, 9(1). https://doi.org/10.1186/s13174-018-0097-0.

Kolbe, T. H., König, G., & Nagel, C. (Eds.). (2011). Lecture Notes in Geoinformation and Cartography. Advances in 3D Geo-Information Sciences. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-12670-3

Kotsev, A., Schleidt, K., Liang, S., van der Schaaf, H., Khalafbeigi, T., Grellet, S., Lutz, M., Jirka, S., & Beaufils, M. (2018). Extending INSPIRE to the Internet of Things through SensorThings API. Geosciences, 8(6), 221. https://doi.org/10.3390/geosciences8060221.

Krämer, M., & Gutbell, R. (2015). A case study on 3D geospatial applications in the web using state-of-the-art WebGL frameworks. In Proceedings of the 20th international conference on 3d web technology (pp. 189–197). https://doi.org/10.1145/2775292.2775303.

Liang, S., Huang, C. & Khalafbeigi, T. (2016). OGC SensorThings API Part 1: Sensing. http://www.opengis.net/doc/is/sensorthings/1.0. Accessed 11 August 2020.

Lim, J., Janssen, P., & Biljecki, F. (2020). VISUALISING DETAILED CITYGML AND ADE AT THE BUILDING SCALE. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLIV-4/W1-2020, 83–90. https://doi.org/10.5194/isprs-archives-XLIV-4-W1-2020-83-2020.

Monsalvete, P., Robinson, D., & Eicker, U. (2015). Dynamic Simulation Methodologies for Urban Energy Demand. Energy Procedia, 78, 3360–3365. https://doi.org/10.1016/j.egypro.2015.11.751.

Moore, V. (2018). Revisiting The Digital Earth. https://cesium.com/blog/2018/01/31/digital-earth-revisited/ Accessed 14 September 2020.

Nouvel, R., Brassel, K. H., Bruse, M., Duminil, E., Coors, V., Eicker, U., & Robinson, D. (2015). SimStadt, a new workflow-driven urban energy simulation platform for CityGML city models. In *Proceedings of International Conference CISBAT 2015 Future Buildings and Districts Sustainability from Nano to Urban Scale* (pp. 889-894).

Schumacher, J. (2020). INSEL 8: Software for simulation, monitoring, and visualisation of energy systems. https://www.insel.eu/en/what-is-insel.html. Accessed 17 August 2020.

T., Santhanavanich C., Kim V., Coors (2020) INTEGRATION OF HETEROGENEOUS CORONAVIRUS DISEASE COVID-19 DATA SOURCES USING OGC SENSORTHINGS API. ISPRS Annals of the Photogrammetry Remote Sensing and Spatial Information Sciences VI-4/W2-2020135-141 10.5194/isprs-annals-VI-4-W2-2020-135-2020

Thunyathep, Santhanavanich Volker, Coors (2021) CityThings: An integration of the dynamic sensor data to the 3D city model. Environment and Planning B: Urban Analytics and City Science 48(3) 417-432 10.1177/2399808320983000

Würstle, P., Santhanavanich, T., Padsala, R., & Coors, V. (2020). The Conception of an Urban Energy Dashboard using 3D City Models. In Proceedings of the Eleventh ACM International Conference on Future Energy Systems (pp. 523–527). ACM. https://doi.org/10.1145/3396851.3402650.

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., Adolphi, T., & Kolbe, T. H. (2018). 3DCityDB - a 3D geodatabase solution for the management, analysis, and visualisation of semantic 3D city models based on CityGML. Open Geospatial Data, Software and Standards, 3(1). https://doi.org/10.1186/s40965-018-0046-7.

Yi, J. S., Kang, Y. A., Stasko, J. T., & Jacko, J. A. (2008). Understanding and Characterising Insights: How do people gain insights using information visualisation?. In Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization (pp. 1-6).

Zirak, M., Weiler, V., Hein, M., & Eicker, U. (2020). Urban models enrichment for energy applications: Challenges in energy simulation using different data sources for building age information. *Energy, 190*, 116292. https://doi.org/10.1016/j.energy.2019.116292.