Towards a multi-database CityGML environment adapted to big geodata issues of urban digital twins

Jean-Paul Kasprzyk [1], Gilles-Antoine Nys [1], Roland Billen [1]
[1] GeoScITY lab, University of Liège, Liège, Belgium – (jp.kasprzyk, ganys, rbillen)@uliege.be

Keywords: GIS, 3DCityDB, Measur3D, CJDB, CityJSON, NoSQL

Abstract

This paper investigates the challenges of implementing the CityGML standard within database environments for managing urban digital twins (UDT), with a particular focus on addressing big geodata issues. CityGML stands as a critical standard for representing and exchanging 3D urban models and thematic data, essential for effective urban planning and infrastructure management. However, integrating CityGML into databases poses challenges due to the dual requirements of geospatial and semi-structured data. While the former imposes a certain rigor in its formalism, the second benefits from more flexibility. Through a comprehensive review and benchmarking of existing database implementations, including both new distributions and those often use, 3DCityDB, CJDB, Measur3D, and Cerbere, this paper proposes a novel approach towards a multi-database CityGML environment tailored to the specific needs of UDT. The proposed solution leverages the strengths of both relational and NoSQL databases, offering a flexible and scalable architecture while ensuring data consistency, geospatial capabilities and compliance with the CityGML schema. The research hypothesis suggests integrating Cerbere, a middleware for CityGML schema compliance and transaction validation, with 3DCityDB and Measur3D (MongoDB). This approach aims to demonstrate the feasibility of managing advanced 3D data operations based on the CityGML model (3DCityDB) and scalability for big data like IoT (Measur3D) within a multi-database environment. The paper contributes to the advancement of UDT by providing a comprehensive solution for managing diverse data types, facilitating more effective urban planning, infrastructure management, and sustainable development initiatives in the context of smart cities.

1. Introduction

In the rapidly evolving landscape of urban development and smart city initiatives, the creation and management of Urban Digital Twins (UDT) have become essential for effective urban planning, infrastructure management, and sustainable growth. One key element in ensuring the success of UDT is the adoption of standardized data formats, with CityGML (Kolbe et al., 2021) standing out as a crucial standard for representing and exchanging 3D city models and their thematic data.

The UDT is a crucial concept in urban planning, bridging the gap between the real world and simulation models. At its core, the UDT uses diverse urban data to create a dynamic representation of the city over time. Among its components, databases play a vital role. These databases store a wide range of information, from infrastructure details to environmental factors, serving as the foundation for decision-making. As cities face the challenges of urbanization, population growth, and sustainability, integrating data from various sources becomes essential (Jeddoub et al., 2023).

However, implementing CityGML within a database involves challenges due to the dual nature of UDT data requirements. While the geospatial information in CityGML aligns well with relational databases (because of their maturity for geodata management), the large and heterogeneous volumes of semi-structured data (big data) are better suited for NoSQL databases and their guiding standards on high availability and scalability. The objective of this paper is to benchmark existing solutions to propose a data server architecture that addresses these challenges and would benefit from the advantages of both modes.

The remaining of this paper is structured as follow. Section 2 is dedicated to the CityGML standard conceptual model and its CityJSON encoding format. Section 3 is a benchmark of previous works related to CityGML implementation within a database. Section 4 describes our research hypothesis based on previously analysed works. Section 5 describes our conclusions and research perspectives.

2. CityGML standard

2.1 CityGML conceptual model

CityGML, an Open Geospatial Consortium (OGC) standard, provides a comprehensive and interoperable conceptual model for storing, and exchanging virtual representations of urban environments. Its importance in the context of UDT implementation cannot be overstated, as it offers several advantages that contribute to the efficiency and effectiveness of the UDT ecosystem: 3D spatial information management, semantic richness, interoperability, multi levels of details (LoD), integration with other standards (e.g. OGC SensorThings API, ISO 19107:2003, etc).

The main idea of CityGML conceptual model (Figure 1) is the management of abstract classes in a core module (e.g. class “AbstractCityObject”) while thematic objects (e.g. a building) are specialized in thematic modules (red modules in Figure 1): buildings, bridges, vegetation, etc. Moreover, all thematic objects are likely to be associated to different concepts described by their own module (blues modules in Figure 1): Appearance, Generics, etc. The CityGML conceptual model is continually evolving, and its latest version, 3.0, introduces opportunities for integrating new sources of big data, such as dynamizers and point clouds (Kutzner, Chaturvedi and Kolbe, 2020).

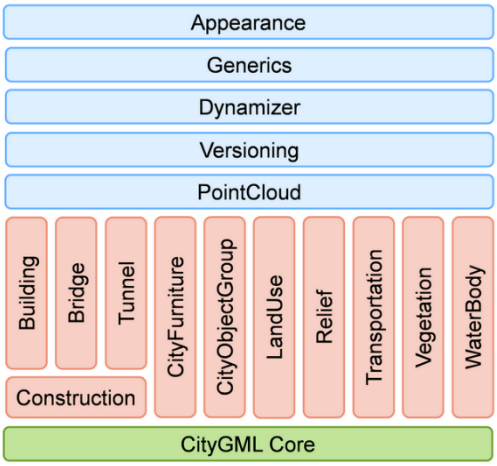

Figure 1. CityGML 3.0 module overview (Kolbe et al., 2021)

2.2 CityJSON format

The format in which data is represented plays a crucial role in its accessibility and usability. While CityGML serves as a comprehensive data model and exchange format for urban environments, its XML-based structure can present challenges for developers due to the complexity of parsers required for navigation. Recognizing this limitation, CityJSON was introduced as a streamlined JSON-based alternative to CityGML by (Ledoux et al., 2019).

CityJSON is designed with programmers in mind, offering rapid software and API development, along with compactness (achieving a sixfold compression compared to CityGML) and compatibility with web applications. This alternative format significantly enhances user-friendliness for both reading and creating datasets, making it a valuable addition to the urban modeling and planning toolkit.

All the possibilities offered by the CityGML data model are not currently available in CityJSON but the differences are reduced and even exceeded on certain points via extensions.

3. Database implementations

3.1 3DCityDB

3DCityDB (Yao et al., 2018) is a relational database management system that implements the CityGML standard (currently v2.0) and provides a wide range of tools around it. Notably, 3DCityDB seamlessly integrates with PostGIS, a spatial database extender for PostgreSQL. This integration enhances spatial capabilities, allowing for robust geospatial operations (including 2D and 3D functions) and analysis within the relational database environment which guarantees CityGML consistency (thanks to constrained tables and attributes). Since its release, 3DCityDB has been used in numerous UDT developments and appears to be the most popular database implementation for 3D urban data management (Dimitrov and Petrova-Antonova, 2021; Diakite et al., 2022; Leopold, Braun and Pinheiro, 2023).

However, 3DCityDB faces significant challenges due to the hierarchical nature of CityGML data. Despite its robust geospatial capabilities and consistency in managing CityGML data, its relational schema complexity is notable. The schema comprises numerous tables (more than 60), necessitating multiple table joins for executing even simple queries. Furthermore, its rigid structure inherent in relational databases can result in numerous null values when handling semi-structured data. Consequently, alternative approaches may be necessary to address the hierarchical nature of UDT data structures more effectively.

3.2 CJDB

In response to the complexity of 3DCityDB database schema, (Powałka et al., 2024) propose a Python-based importer/exporter alternative, called "CJDB", for storing CityGML data in PostgreSQL. It results in only 3 tables derived from UML class diagram shown in Figure 2:

- a table "city_object", storing all attributes of CityGML abstract class "CityObject".
- a table "city_object_relationships", storing the possible parent-children relationships between rows of "city_object" table (e.g. a building part like a wall is a children of a building).
- a table "cj_metadata", storing metadata related to a city model (i.e. a set of city objects)

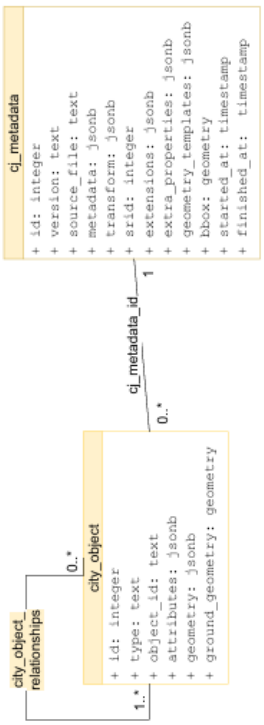

Figure 2. UML diagram of CJDB (Powalka et al., 2023)

The main idea behind this simplified schema is to manage the complexity of CityGML instanced objects within JSON attributes. For example, all the attributes of a thematic object (e.g. a building) are stored as a JSON compliant with CityJSON subschema related to this object (within column "attributes" of table "city_object").

Table "city_object" also includes a column "geometry" of JSON type. It stores the CityGML geometry of a city object as a JSON similar to its CityJSON definition (an array of JSON because multiple geometries can be attached to a single object at distinct Levels of Details). The difference with CityJSON lies in the management of coordinates which are directly stored in attribute "boundaries" in CJDB while CityJSON uses attribute "boundaries" as a set of indexes referencing to vertices whose

coordinates are stored separately. In other words, CityJSON allows vertices to be shared between CityObjects, consequently storing their topology and avoiding redundancy, which is not the case in CJDB.

Thanks to its JSON storage and simplified schema, CJDB appears as a relevant solution to easily manage complex CityGML data in relational database, especially for client applications dealing with CityJSON. However, we note two inconveniences. Firstly, the storage of 3D geometries within a JSON column makes it difficult to perform 3D spatial queries within the database. Indeed, an on-the-fly conversion from JSON geometries to PostGIS geometries is necessary to benefit from PostGIS 3D functions. Secondly, the consistency of CityGML schema is not guaranteed since PostgreSQL does not support additional constraints to its JSON type. Despite its complexity and its rigid structure, 3DCityDB is more consistent and directly benefits from PostGIS 3D functions.

3.3 Measur3D and Cerbere

In addition to 3D geodata, UDT also include various sources of thematic data and especially Internet of Things (IoT) data: traffic cameras, air quality sensors, weather stations, energy monitoring of buildings, etc. These data are characterized by very large volumes and heterogeneous structures which are not adapted to relational databases. Thanks to their more flexible schemas and scalability, NoSQL databases are more suited to these big data (Moniruzzaman and Hossain, 2013). Therefore, (Nys and Billen, 2021) propose a CityGML implementation within a document-oriented database: MongoDB. This implementation is called “Measur3D”.

In a document-oriented database, records are stored as documents that adhere to schemas that are non-mandatory and semi-structured (i.e. schemas might be predefined but also might be modified as time goes on). Documents sharing a common semi-opened schema are gathered in a collection so that documents inherit from their collection attributes. (Nys and Billen, 2021) propose 5 collections to manage CityGML data, including:

- “CityModel” (metadata of the city model)
- “AbstractCityObject” (referring to the corresponding CityGML abstract class)
- “Geometry” (as defined by CityJSON format)
- “Texture” and “Material” (used for model appearances purpose)

As CJDB implementation, the abstract city object class has a pre-defined structure while children thematic objects (buildings, vegetation, etc) are stored in BSON (binary JSON) documents following CityJSON schema without anyconstraints and so no guaranteed consistency. To handle this issue, (Nys and Billen, 2023) propose a middleware called “Cerbere” between the database and the client applications. The entire CityGML schema is implemented in Cerbere so that every transaction is validated to maintain the database compliance with CityGML. This solves the consistency issue of simplified CityGML schema while benefiting from scalability and flexibility of the NoSQL database. However, 2D spatial functions are still quite limited in MongoDB and 3D spatial functions non-existent.

The architectural flexibility of Cerbere system is highlighted by its ability to accommodate variations in schema definitions without necessitating modifications to either the initial requirements, either the database schema or the data already present in the database. Acting as a middleware, the system filters information between the server and client applications, ensuring that only correctly structured and relevant information is stored and retrieved from the document-oriented database. Through bidirectional filtering, the system maintains consistency in both query requests and database responses, enforcing schema specifications to validate semantic information on-the-fly.

For instance, Figure 3 scenario depicts two applications seeking to register and retrieve information about the same object from the same class. With both applications utilizing identical unique identifiers, whether they are URIs (Uniform Resource Identifiers), UUIDs (Universally Unique Identifiers), etc., the object is defined by four attributes: two handled by each application. Notable observations include: attribute_2 is excluded from storage in the database due to its prohibition in the server schema of the first application. Consequently, the information fails the writing filter and is not transmitted to the database, rendering it unqueryable thereafter. Similarly, attribute_4 is omitted from storage as it fails to conform to the formatting requirements of the second application's schema. If correctly formatted, it would be stored in the database and accessible to clients. Conversely, attribute_1 and attribute_3 are stored in the database, but their utility is confined to the respective frameworks of the two applications.

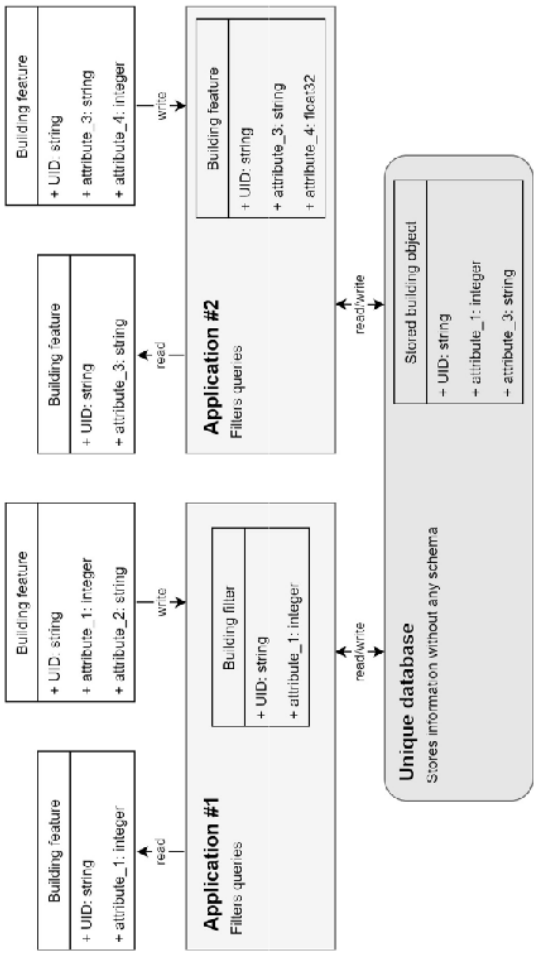

Figure 3. Example of the bi-directional filter principles of Cerbere (Nys and Billen, 2023)

This middleware serves a dual purpose: ensuring consistency and delivering relevant information to each application. Unlike the database, which merely acts as a storage repository without

guaranteeing data integrity, developers play a critical role in enforcing consistency through their code. The database, in this context, serves as the central hub for all data consumers—clients, applications, and more. This approach emphasizes extreme flexibility, in stark contrast to the relational model, where middleware is absent, and integration must be meticulously structured in advance.

4. Hypothesis

As will be recalled the objective of this research is the development of a data server architecture which addresses the database implementation issue of CityGML in a UDT context. Considering the literature review, we identified three solutions (four if we consider Measur3D with and without Cerbere) and compared their characteristics in Figure 4:

- 3DCityDB benefits from PostGIS 3D geodata management and consistency but it leads to a complex relational schema not suited for big data issues and dynamic changes.
- CJDB significantly reduces the complexity of 3DCityDB and makes CityJSON export easier for applications but loses consistency and makes it difficult to use PostGIS 3D functions.
- Measur3D proposes a simplified schema close to CJDB but stores it in a document-oriented database (MongoDB) more suited for big data. The resulting consistency issue is solved by middleware Cerbere which validates every transaction regarding CityGML schema.

	3DCityDB	CJDB	Measur3D	Measur3D + Cerbere
Consistency				
Scalability				
Flexibility				
2D geospatial functions				
3D geospatial functions				
CityJSON import/export				

Figure 4. Comparison of CityGML database implementations in a UDT context: green is better than orange and orange is better than red

Our hypothesis is that Cerbere could be connected to these three systems to maintain CityGML consistency and allow interrogation of different databases with a single API (Figure 5).

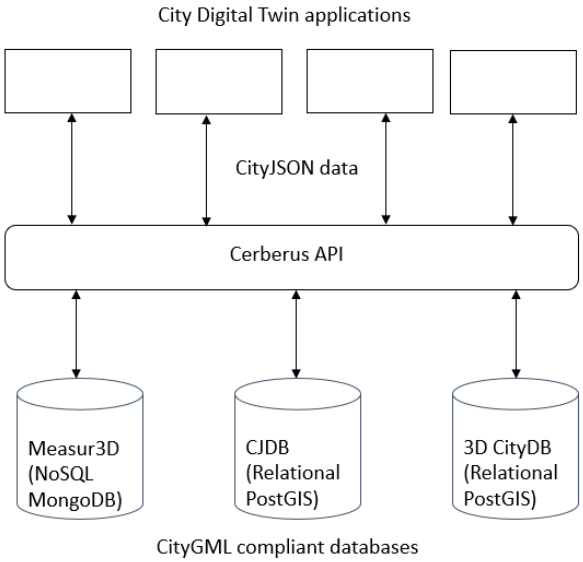

Figure 5. CityGML data server suited for big data issues related to urban digital twins.

This approach could facilitate the integration of NoSQL databases in a UDT architecture while still benefiting from complex 3D operations in relational GIS. Queries from various applications would be sent to Cerbere which would transmit it to the right database(s), depending on the nature of asked information (e.g. NoSQL database for IoT data) and the spatial complexity of the query (3DCityDB for 3D geoprocessing). Indeed, a proof of concept showing traditional GIS and NoSQL database interactions has already been proposed by (Holemans, Kasprzyk and Donnay, 2018).

This should be seen as an alternative to the already widely used GraphQL, which provides only a single endpoint and only handles queries sequentially.
The next step of this research is to extend Cerbere API to 3DCityDB to implement a UDT proof of concept in a hybrid solution including Measur3D for semantic aspect and 3DCityDB for spatial aspect.

5. Conclusion and perspectives

In conclusion, this paper addresses the critical challenge of implementing the CityGML data model within a database environment optimized for managing the complexities of urban digital twins (UDT). It particularly focuses on the issues posed by big geodata: the large amount of data, the heterogeneity of their sources and their high frequency of creation and update. CityGML, as a standardized format, plays a pivotal role in facilitating the representation and exchange of 3D urban models and thematic data. However, integrating CityGML into databases involves navigating the dual requirements of geospatial and semi-structured data.

Through a comprehensive review of existing solutions, this paper proposes a novel approach towards a multi-database CityGML environment tailored to the specific needs of UDT. The benchmarking of database implementations, including 3DCityDB, CJDB, Measur3D, and Cerbere, highlights their respective strengths and limitations. While relational databases like 3DCityDB offer robust geospatial capabilities and consistency, they struggle with scalability and handling semi-

structured data. On the other hand, document-oriented NoSQL databases like MongoDB are better suited for big data but lack sophisticated spatial functions and struggle with maintaining data consistency.

The proposed solution leverages the strengths of both relational and NoSQL databases while mitigating their shortcomings. By integrating a middleware like Cerbere, which ensures CityGML schema compliance and transaction validation, with various database implementations, it becomes possible to maintain data consistency while benefiting from the scalability and flexibility of NoSQL databases. This approach enables seamless interrogation of different databases through a single API, facilitating the integration of IoT data and complex 3D operations within the UDT architecture.

Looking forward, this paper presents a research hypothesis that lays the groundwork for future investigations in the realm of urban digital twins and database management. As the field continues to evolve, there remains many opportunities to refine and expand upon the proposed solutions outlined herein.

A key perspective for future research involves conducting a proof of concept by connecting 3DCityDB, known for its advanced 3D data management and processing capabilities based on the CityGML model, with MongoDB (Measur3D) through Cerbere. This endeavor aims to demonstrate the feasibility of integrating both relational and NoSQL databases within a single environment. By leveraging the strengths of each database type, such as 3DCityDB's spatial operations and MongoDB's scalability for IoT data, this integrated approach could offer a comprehensive solution for managing diverse data types in UDT scenarios.

Acknowledgements

We gratefully acknowledge the “Conseil Universitaire à la Recherche et à la Valorisation” (CURV) of the University de Liège for funding and supporting the project “Cerbere” including this research. We appreciate CURV's commitment to promoting excellence and facilitating impactful research endeavours at our institution.

This research is part of the project GIS 3.0 that demonstrates the convergence of Geographic Information Systems and Web 3.0: Semantic Web techniques, object-oriented prototype languages (JavaScript, JSON,) and document-oriented NoSQL databases. The research project (PDR) is funded by the Belgian National Funds for Scientific Research FNRS_2019_SIG3.0_PDR/OL T.0024.20.

References

Diakite, A. *et al.* (2022) ‘FINAL REPORT (RG202877) Liveable City Digital Twin: Analytics for agile decision making’.

Dimitrov, H. and Petrova-Antonova, D. (2021) ‘3D CITY MODEL AS A FIRST STEP TOWARDS DIGITAL TWIN OF SOFIA CITY’, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B4-2021, pp. 23–30. Available at: https://doi.org/10.5194/isprs-archives-XLIII-B4-2021-23-2021.

Holemans, A., Kasprzyk, J.-P. and Donnay, J.-P. (2018) ‘Coupling an Unstructured NoSQL Database with a Geographic Information System’, in C.-P. Rückemann and Y. Doytsher (eds) *GEOProcessing 2018 : The Tenth International Conference on Advanced Geographic Information Systems, Applications, and Services*. *GEOProcessing 2018 : The Tenth International Conference on Advanced Geographic Information Systems, Applications, and Services*, Rome, Italy: IARIA, pp. 23–28. Available at: http://hdl.handle.net/2268/221769.

Jeddoub, I. et al. (2023) ‘Digital Twins for cities: Analyzing the gap between concepts and current implementations with a specific focus on data integration’, International Journal of Applied Earth Observation and Geoinformation, 122, p. 103440. Available at: https://doi.org/10.1016/j.jag.2023.103440.

Kolbe, T.H. *et al.* (2021) ‘OGC City Geography Markup Language (CityGML) Part 1: Conceptual Model Standard’. Open Geospatial Consortium. Available at: http://www.opengis.net/doc/IS/CityGML-1/3.0.

Kutzner, T., Chaturvedi, K. and Kolbe, T.H. (2020) ‘CityGML 3.0: New Functions Open Up New Applications’, PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science, 88(1), pp. 43–61. Available at: https://doi.org/10.1007/s41064-020-00095-z.

Ledoux, H. *et al.* (2019) ‘CityJSON: a compact and easy-to-use encoding of the CityGML data model’, *Open Geospatial Data, Software and Standards*, 4(1), p. 4. Available at: https://doi.org/10.1186/s40965-019-0064-0.

Leopold, U., Braun, C. and Pinheiro, P. (2023) ‘AN INTEROPERABLE DIGITAL TWIN TO SIMULATE SPATIO-TEMPORAL PHOTOVOLTAIC POWER OUTPUT AND GRID CONGESTION AT NEIGHBOURHOOD AND CITY LEVELS IN LUXEMBOURG’, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLVIII-4/W7-2023, pp. 95–100. Available at: https://doi.org/10.5194/isprs-archives-XLVIII-4-W7-2023-95-2023.

Moniruzzaman, A.B.M. and Hossain, S.A. (2013) ‘NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison’. Available at: https://doi.org/10.48550/ARXIV.1307.0191.

Nys, G. and Billen, R. (2021) ‘From consistency to flexibility: A simplified database schema for the management of CityJSON 3D city models’, *Transactions in GIS*, 25(6), pp. 3048–3066. Available at: https://doi.org/10.1111/tgis.12807.

Nys, G. and Billen, R. (2023) ‘From consistency to flexibility: Handling spatial information schema thanks to a middleware in a 3D city modeling context’, *Transactions in GIS*, 27(1), pp. 115–133. Available at: https://doi.org/10.1111/tgis.13014.

Powałka, L. *et al.* (2023) ‘cjdb: a simple, fast, and lean database solution for the CityGML data model’. Available at: https://doi.org/10.48550/ARXIV.2307.06621.

Powałka, L. et al. (2024) ‘cjdb: A Simple, Fast, and Lean Database Solution for the CityGML Data Model’, in T.H. Kolbe, A. Donaubauer, and C. Beil (eds) Recent Advances in 3D Geoinformation Science. Cham: Springer Nature Switzerland (Lecture Notes in Geoinformation and Cartography), pp. 781–796. Available at: https://doi.org/10.1007/978-3-031-43699-4_47.

Yao, Z. *et al.* (2018) ‘3DCityDB - a 3D geodatabase solution for the management, analysis, and visualization of semantic 3D city

models based on CityGML', *Open Geospatial Data, Software and Standards*, 3(1), p. 5. Available at: https://doi.org/10.1186/s40965-018-0046-7.