

# Semantic 3D City Database – an enabler for a dynamic geospatial knowledge graph

Arkadiusz Chadzynski<sup>1</sup>, Nenad Krdzavac<sup>1</sup>, Feroz Farazi<sup>2</sup>, Mei Qi Lim<sup>1</sup>,  
Shiyong Li<sup>4</sup>, Ayda Grisiute<sup>4</sup>, Pieter Herthogs<sup>4</sup>, Aurel von Richthofen<sup>4</sup>,  
Stephen Cairns<sup>4</sup>, Markus Kraft<sup>1,2,3</sup>

released: June 1, 2021

<sup>1</sup> CARES  
Cambridge Centre for Advanced  
Research and Education in Singapore  
1 Create Way  
CREATE Tower, #05-05  
Singapore, 138602

<sup>2</sup> Department of Chemical Engineering  
and Biotechnology  
University of Cambridge  
Philippa Fawcett Drive  
Cambridge, CB3 0AS  
United Kingdom

<sup>3</sup> School of Chemical  
and Biomedical Engineering  
Nanyang Technological University  
62 Nanyang Drive  
Singapore, 637459

<sup>4</sup> Singapore-ETH Centre  
at CREATE  
1 Create Way  
CREATE Tower, #06-01  
Singapore, 138602

Preprint No. 273



---

*Keywords:* CityGML, Sustainability, Digitisation, Urban Planning, Semantic Web, Knowledge Graph, Ontology, Decision Support System, Artificial Intelligence, Geospatial Modelling, Geospatial Search

**Edited by**

Computational Modelling Group  
Department of Chemical Engineering and Biotechnology  
University of Cambridge  
Philippa Fawcett Drive  
Cambridge CB3 0AS  
United Kingdom

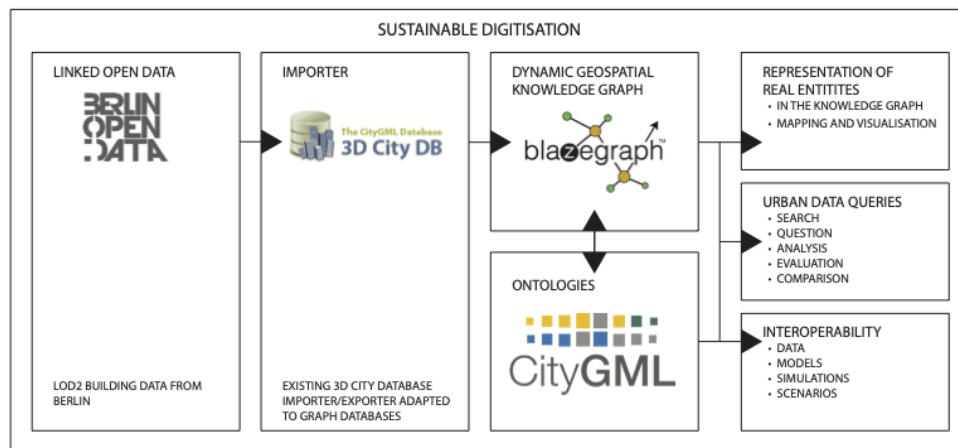
**E-Mail:** [mk306@cam.ac.uk](mailto:mk306@cam.ac.uk)

**World Wide Web:** <https://como.ceb.cam.ac.uk/>



## Abstract

This paper presents a dynamic geospatial knowledge graph as part of The World Avatar project, with an underlying ontology based on CityGML 2.0 for three-dimensional geometrical city objects. We comprehensively evaluated, repaired and refined an existing CityGML ontology to produce an improved version that could pass the necessary tests and complete unit test development. A corresponding data transformation tool, originally designed to work alongside CityGML, was extended. This allowed for the transformation of original data into a form of semantic triples. We compared various scalable technologies for this semantic data storage and chose Blazegraph™ as it provided the required geospatial search functionality. We also evaluated scalable hardware data solutions and file systems using the publicly available CityGML 2.0 data of Charlottenburg in Berlin, Germany as a working example. The structural isomorphism of the CityGML schemas and the OntoCityGML Tbox allowed the data to be transformed without loss of information. Efficient geospatial search algorithms allowed us to retrieve building data from any point in a city using coordinates. The use of named graphs and namespaces for data partitioning ensured the system performance stayed well below its capacity limits. This was achieved by using scalable and dedicated data storage hardware capable of hosting expansible file systems, which strengthened the architectural foundations of the target system.



## Highlights

- OntoCityGML based on CityGML 2.0 and W3C standards.
- Architecture definition for a dynamic geospatial knowledge graph enabled by the Semantic 3D City Database.
- Data interoperability capabilities provided by means of sustainable digitisation practices.

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Refined Ontology for CityGML</b>	<b>5</b>
2.1	An evaluation of the CityGML ontology . . . . .	5
2.2	An evaluation of the OntoCityGML ontology . . . . .	6
<b>3</b>	<b>Augmented Data Transformation Tools</b>	<b>7</b>
3.1	3D City Database Importer/Exporter Tool . . . . .	7
3.2	Relational Schema to Graph Mapping . . . . .	9
<b>4</b>	<b>Semantic 3D City Data Store</b>	<b>12</b>
<b>5</b>	<b>Hardware Requirements</b>	<b>15</b>
<b>6</b>	<b>Conclusions and future work</b>	<b>19</b>
<b>A</b>	<b>OntoCityGML knowledge-base expressed in a DLs syntax</b>	<b>22</b>
<b>B</b>	<b>Sample of unit test cases - covering CityModelType term in OntoCityGML</b>	<b>27</b>
	<b>References</b>	<b>29</b>

# 1 Introduction

## General context of the paper - problem space

Development of sustainable digitisation practices is widely recognised as an important part of roadmaps at organisational, industry [60], national [3] as well as international levels [31]. Radermacher [50] points to the fact that global governing bodies, such as the UN, G20 and the World Bank, all agree on the importance of adopting digitisation standards for achieving international comparability. Despite the complexity of existing standards and the time-consuming adoption and implementation of information systems into a digital form, those bodies agree that the benefits greatly outweigh the costs. Radermacher [50] also notices that a scientific approach and adherence to such standards ensures trust based on evidence. Moreover, it also positively impacts accountability, transparency and control of execution at all levels. Undoubtedly, roadmaps strengthening comparability include technological solutions designed with intention to support systems interoperability as well [51].

The World Avatar (TWA) is an all-encompassing dynamic knowledge graph. It is built on agent-based system [3, 52] architectural principles. Within it, intelligent agents operate on a knowledge graph built in accordance with semantic web standards and recommendations provided by the W3C. Therefore, the system can be regarded as an example of a general knowledge graph, capable of multi-domain knowledge representation [3, 17, 18, 20, 22, 23, 41, 64, 65]. Answering to inter-domain interoperability problems is at its core. The target system is a digital representation of the world. Representations of different domains must also adhere to applicable standards.

Designed as one of the critical TWA components, the J-Park Simulator (JPS) [35, 47, 48, 62, 63, 66] includes representations of built environments and agent-based subsystems capable of simulating emissions dispersion from various types of air pollution sources as well as optimising designs of Eco-Industrial Parks (EIPs) with respect to their carbon footprint [21].

The system architecture for the Semantic 3D City Database proposed in this paper aims at closing some of the gaps, particularly related to current built environment representation within JPS and TWA. Basing it on reusable Open Source components and standardised interfaces encourages wider adoption throughout other information systems that require scalable and interoperable three-dimensional representation of such environments.

## Cities and geospatial information

Existing City Information Models (CIM) already integrate large urban datasets in order to represent multiple aspects of cities such as the built environment, energy management, transport, etc [26]. Linking across domains and securing scalability are key challenges to developing urban Digital Twins (DT) [53]. Representing cities as three-dimensional models of built environments is a crucial step to add more urban data and knowledge [59] to representing urban environments that are developed and planned.

One of the common ways of 3D modelling for built environments in various information systems is to use the CityGML standard, provided by the Open Geospatial Consortium (OGC) [28]. This can be used as a data exchange standard for city landscape management

and planning systems or even as a file-based data source for applications visualising 3D city landscapes on the web. It is possible to encode information about different domains within this format through domain-specific extensions as well as combine purely geospatial concerns with any others in order to analyse or support decisions regarding digitised urban design blueprints. A digital twin of the Manchester landscape in CityGML 2.0, with solar irradiation projected on the roofs of the buildings [43], is just one of the plethora of examples currently available on the web. However, developing applications built on static files lacks flexibility. Apart from compliance with standards, flexibility is a key ingredient to achieving interoperability [51]. It also keeps standard based systems open to future innovation. Dynamism of the stored data is required to be able to perform simulations under various conditions and hot swap certain information in representations on demand. Moreover, dynamic representation allows the gaps in static city models arising from the constant evolution of the entities within built environments to be addressed.

The open source 3D City Database, developed at the Technische Universität München (TUM), was meant to close some of those gaps [57]. 3D City Database is a suite of tools to transform data encoded in flat CityGML 2.0 files into a more flexible database format and to store and visualise 3D city landscapes. It has been under development since 2003 [61]. The flagship examples, which showcase how to store and visualise city data with the help of those tools, are models of Berlin and New York in Level Of Detail 2 (LOD2). This approach demonstrates a possibility of storing city data and adhering to the CityGML 2.0 standard in a different way than by using static XML files. However, relational database backends of this solution, limit implementation of semantic data interoperability. While some authors have reported on first attempts to use graph databases for geospatial data [2], discussions contain predominantly general ideas and partial results [46]. Adopting a semantic data store allows for turning a bare 3D City Database into a knowledge base [52] with inference, truth maintenance and reasoning engines. This makes the resulting Semantic 3D City Database an enabler for the dynamic geospatial knowledge graph in TWA.

## Synthesis

Research at the University of Geneva focused on the other side of the problem spectrum led to producing CityGML ontology. It turned out to be possible to generate an ontology, with one to one matching between concepts, by applying XSLT transformations into original CityGML 2.0 schemas. The ontology produced by applying those techniques could be regarded as a step towards bringing the standard and applications providing semantic interoperability together. However, closer examination of the available ontology reveals a number of issues concerning its quality and, because of that, suitability to be used as a Tbox (an ontology schema) for reliable applications adhering to semantic web standards and recommendations. Apart from that, there is also a lack of data transformation tools which would allow population of the ontology with data and produce instances for an Abox. Schema and instances are equally needed to build any application able to operate a three-dimensional geospatial data. Such applications also need to provide reliable geospatial search functionality with acceptable data retrieval times [38]. Although there are semantic triple stores implementing geospatial search, there is a lack of examples of semantic web applications operating on the multitude of geometries required to represent entire cities. Any software application satisfying such requirements needs appropriate

hardware to facilitate this specific functionality. Fully semantic 3D city database architecture definitions, bringing together all the above specified components in order to provide foundations for semantic applications operating on dynamic city models, are not currently available.

The **purpose of this paper** is to present such an architecture definition as well as the steps necessary to produce proof of concept solutions that address the mentioned problems. The ontology refinement process and its evaluation with regard to quality and correctness, required to ensure Tbox reliability, are elaborated on in the following Section 2. Next, the use of refined ontology concepts in the process of augmenting data transformation tools is presented, based on existing open source solutions (Section 3). Evaluation of existing semantic data stores, carried on before producing an Abox by utilising terms of the refined ontology, is described in Section 4. The final Section 5 is devoted to presenting estimated hardware requirements, as a result of evaluating geospatial functionality on tens of thousands of buildings with over 2000 different types of two-dimensional and three-dimensional geometrical shapes.

## 2 Refined Ontology for CityGML

Producing a proof of concept Semantic 3D City Database required ensuring reliability of the ontology, used as its schema, in the first step. As a result, following the general principle of reusability, such schema – OntoCityGML – is based on an existing ontology reflecting the CityGML 2.0 standard and developed at the University of Geneva [15, 36]. Because of the methods used to produce it, it is referred to as CityGML ontology in the following subsections, where the steps and methodology undertaken in the process of refining it to a version suitable for the proof of concept solutions are elaborated on.

### 2.1 An evaluation of the CityGML ontology

The publicly available CityGML ontology [15], which served as a base for OntoCityGML ontology, was developed as a result of applying XSLT transformations to CityGML 2.0 schema [28] as well as some manual mapping needed to generate it [15]. The ontology implements 185 classes, 281 object properties and 92 data properties. It also contains 1254 implemented axioms. Categorisation of criteria used during its evaluation in order to check suitability for the proof of concept presented in this paper is enumerated in Table 1. The suite of tools and plugins available in the Protégé ontology editor [45] was utilised during this process.

The following errors were reported by those tools in the CityGML ontology, and then manually fixed during evaluation, based on the all of the above metrics. First, it did not pass the *Accuracy* test. There were a number of *Illegal redeclarations of entities: reuse of entity* errors. For instance, the term *year of construction*, also present in the original Charlottenburg CityGML 2.0 data, was implemented as *owl:ObjectProperty* and *owl:DatatypeProperty* entities at the same time. More than fifty errors of this nature were reported by the editor. Second, the *Conciseness* test checked whether the base on-

**Table 1:** *Categorisation of Ontology Evaluation Criteria (updated from [29]).*

Evaluation perspective	Metrics
Ontology correctness	Accuracy
	Conciseness
	Completeness
	Consistency & Coherence
Ontology quality	Computation efficiency

tology defined entities which were irrelevant to the domain to be covered [29]. Conducting it on the CityGML ontology revealed entities which were irrelevant as ontology elements with regard to TWA domain. For example, the term *class* was implemented as *owl:ObjectProperty* for entities which were irrelevant to TWA domain. The base ontology contained more than ten entities of this nature. Third, the *Completeness* test measured whether the domain of interest was appropriately covered [29]. It revealed that the CityGML ontology did not completely cover TWA domain. For instance, a term such as *envelope*, essential for the proof of concept, was defined in the CityGML 2.0 open data model but was not implemented in the CityGML ontology [15]. Fourth, the HermiT [27] reasoner revealed the *Coherence* and *Consistency* of the CityGML ontology. Finally, the *Computational efficiency* test showed that the expressivity of Description Logics (DLs) of the CityGML ontology was equivalent to  $\mathcal{AL}\mathcal{E}\mathcal{H}\mathcal{I}(\mathcal{D})$  DLs. The described assessment ensured that the CityGML ontology could serve as a good starting point for the development of an OntoCityGML ontology and provide some foundations for creating a dynamic geospatial knowledge graph.

## 2.2 An evaluation of the OntoCityGML ontology

OntoCityGML ontology, which served as a Tbox [5] for the proof of concept Semantic 3D City Database, is an extension of the CityGML ontology and a result of resolving previously mentioned issues. As in the case of the base ontology, tools and plugins available for the Protégé ontology editor [45] were used for its evaluation. The OntoCityGML ontology implements 344 classes, 272 object properties and 78 datatype properties. It also contains 3363 implemented axioms.

The *Computational efficiency* test shows that the expressivity of OntoCityGML ontology DLs is equivalent to  $\mathcal{AL}\mathcal{E}\mathcal{H}(\mathcal{D})$  DLs. Due to the DLs' expressivity of the ontology falling between DL-Lite [11] and  $\mathcal{SR}\mathcal{O}\mathcal{I}\mathcal{Q}$  DL [30], the OntoCityGML cannot be used to query city data stored in relational databases by the means of the ontology-based data access (OBDA) technologies [39]. The HermiT reasoner is able to classify the OntoCityGML ontology. In debugging mode, it also detects that this ontology is *Consistent* and *Coherent*. The OntoCityGML ontology fully passed *Accuracy*, *Conciseness* and *Completeness* tests as well. Protégé does not show any errors related to illegal declaration of entities or reuse of entities.

To cover TWA domain appropriately and to the extent needed for the proof of concept, sixty-nine new terms were implemented into the OntoCityGML ontology. The terms were



checked for one to one correspondence between the implementation in the ontology and the CityGML 2.0 specification [28]. The list of terms corresponds to the unique list of CityGML 2.0 tags found in the Charlottenburg-Wilmersdorf data used in this proof of concept. OntoCityGML axioms relevant to this list are included in the Appendix A.

Additionally, each of the new terms implemented in the OntoCityGML was covered by unit tests. This step ensures that any further changes to the OntoCityGML ontology preserve its structure. Sample test cases are included in Appendix B. Furthermore, the OntoCityGML ontology has been placed under git version control system in order to make tracking such changes transparent.

### 3 Augmented Data Transformation Tools

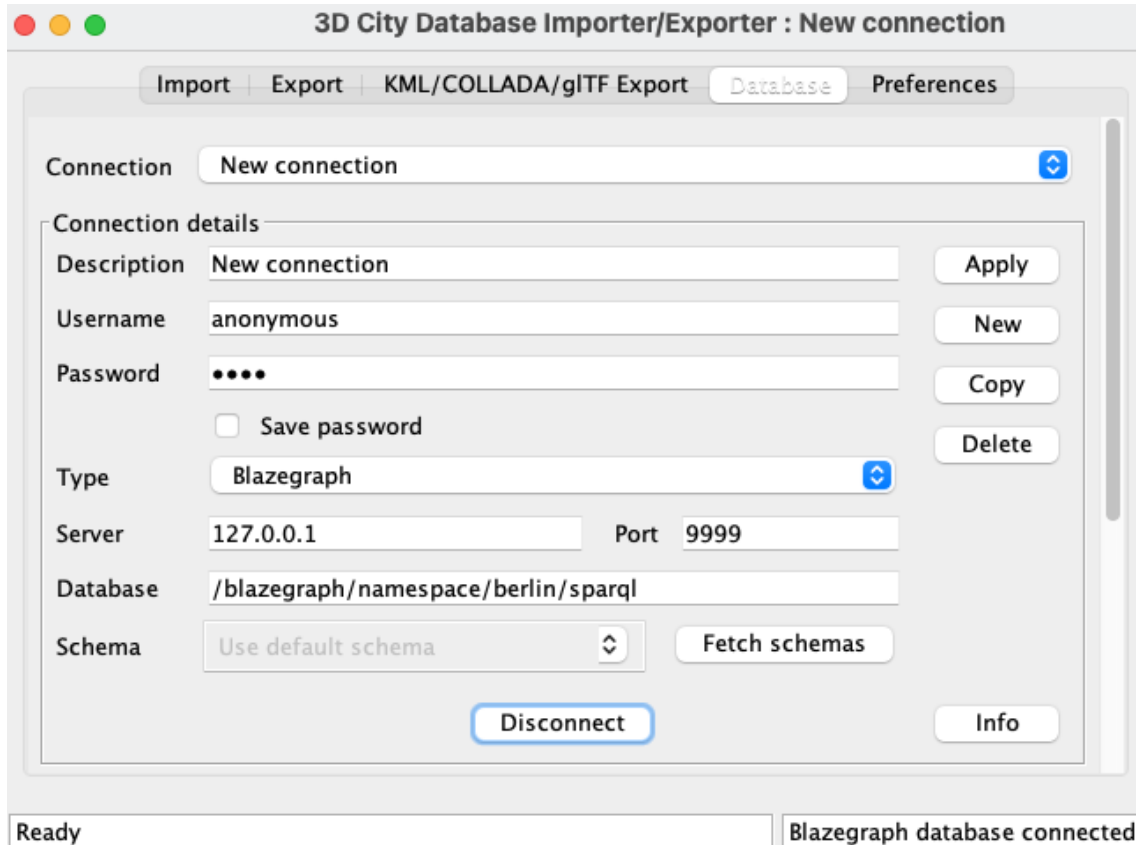
For the proof of concept, data validation mechanisms of the augmented Importer/Exporter tool, originating from TUM [57], were used to transform CityGML 2.0 data to the Semantic 3D City Database [5] which uses OntoCityGML terms to describe city models. In this process, every CityGML object is validated by the tool prior to its instantiation to a corresponding Java object. The data transformation process is described in the next section (3.1).

#### 3.1 3D City Database Importer/Exporter Tool

Depending on the level of detail, CityGML models can form quite complex and, when measured by the average present-day computing capabilities, relatively large datasets. An architecture of any application designed to work with such models needs to be developed with this in mind. In the particular case of the semantic geospatial knowledge graph, it has to ensure efficiency of SPARQL queries on an Abox as well as balance performance and optimal data storage. While citygml4j [16], an open source Java class library and API, provides a very good start to work with CityGML 2.0 models programmatically, there is a lack of tools which would be able to turn such models into semantic triples forming an Abox of a geospatial knowledge graph.

After exploring options, the closest existing data transformation tool able to fulfill such requirements is the 3D City Database Importer/Exporter [1]. It is also based on citygml4j and is available as an open source project. The TUM tool is optimised to work with large CityGML 2.0 models and uses multithreading to read the data, transform it and write it into a database [57]. Hence, it is more computationally efficient than a raw library when the potential number of a city model objects being processed simultaneously is taken into consideration. This matters in the case of large and detailed models. The unmodified tool supports Oracle and PostGIS relational databases and makes use of the Java Database Connectivity (JDBC) API with respective database connectors. This particular design allowed the reuse and augmentation of large parts of its code to work with Semantic 3D City Database based on a non-relational graph triple store.

In order to augment the tool in such a way, Jena JDBC, A SPARQL over JDBC driver framework [32], was utilised for the proof of concept presented in this paper, with choos-



**Figure 1:** *CityGML 2.0 data transformation tool augmented to support Blazegraph™ as a data store back-end. In the depicted menu, connection to the semantic database was established and the tool is ready to start importing the data. In this augmented version of the Importer/Exporter tool [57], city model data will be imported via executing SPARQL statements with OntoCityGML vocabulary against the semantic data store, instead of SQL statements against a relational database with predefined 3DCityDB schema, like in the original version. The original functionality is preserved and relational database types can still be used.*

ing its Remote Endpoint driver connecting into a SPARQL Protocol compliant triple store that exposes SPARQL query and SPARQL update endpoints. Adding the new driver allowed augmentation of the tool and preservation of its original functionality. The majority of the new features were added into two of the tool’s original code packages: *impexp-client* and *impexp-core*. While modifying the first one allowed addition of components enabling the selection of new database type, augmentation of the second component allowed the tool to be capable of establishing a connection to a triple store via JDBC. This required adding a new database backend adapter and incorporation of five new classes into the foundation codebase. Namely, *BlazegraphAdapter*, *GeometryConverterAdapter*, *SchemaManagerAdapter*, *SQLAdapter* and *UtilAdapter* classes had to be implemented, at minimum, for the tool to be able to facilitate connectivity to a semantic triple store with the new driver.

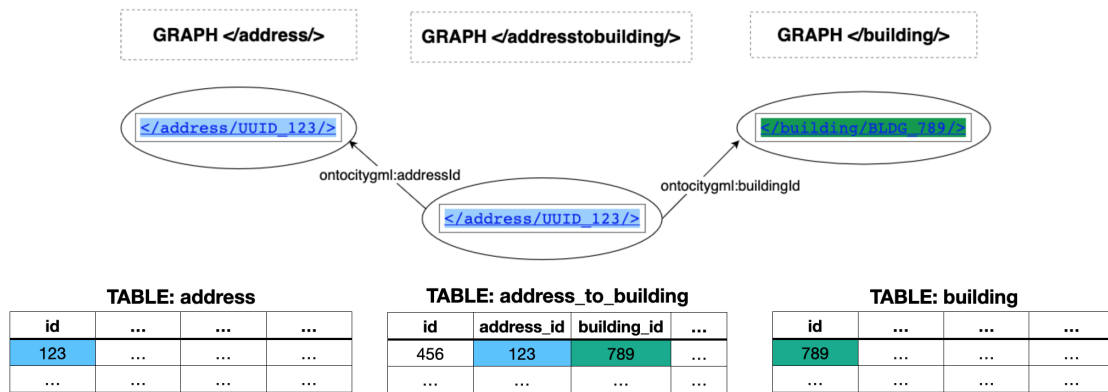
As the Importer/Exporter tool was originally designed to work with relational databases at a very high level, it was validating the CityGML models before instantiating model members into Java objects, which were, in turn, persisted in a database by the means of corresponding SQL statements. For the tool to be fit for purpose when used as a data transformation tool with the Semantic 3D City Database, the last step had to be augmented with the functionality to generate equivalent SPARQL statements for the respective Java objects. Preserving current data structures leveraged many years of development spent on query and storage optimisation at TUM while developing it and fine-tuning over that time [57].

In order to produce a semantic twin of the 3D City Database representation for the Charlotteburg-Wilmersdorf district of Berlin, the following classes of the *org.citydb.citygml.importer.database.content* module had to be modified to work with SPARQL JDBC prepared statements instead of SQL JDBC prepared statements: *DBCityObject*, *DBBuilding*, *DBAddress*, *DBAddressToBuilding*, *DBCityObjectGenericAttrib*, *DBSurfaceGeometry*, *DBAppearance*, *DBAppearToSurfaceData*, *DBExternalReference*, *DBSurfaceData*, *DBTexImage*, *DBTextureParam*, *DBThematicSurface*. New methods generating SPARQL prepared statements were added to each of these classes and covered by appropriate unit tests. The existing code was augmented to fill in those statements with CityGML objects data when the semantic backend is specified as a chosen option for the 3D City Database Importer/Exporter tool. As depicted in Figure 1, the modified tool is able to produce the Semantic 3D City Database Abox, using OntoCityGML as a Tbox. It also produces a semantic “mirror twin” of the relational 3D City Database, with additional properties specific to semantic knowledge bases. Those properties are described in more detail in the next subsection.

## 3.2 Relational Schema to Graph Mapping

The heart and soul of much mathematics consists of the fact that the “same” object can be presented to us in different ways [44]. The present section elucidates this statement while taking into consideration the results of data transformations, which are outcomes of the augmented data transformation tool. The augmented Importer/Exporter is able to produce the original database as well as the Semantic 3D City Database. Structural isomorphism of the 3D City Database and its semantic twin, illustrated in Figure 2, shows their equivalence.

There is one to one correspondence [6] between schemas of the proof of concept databases, in terms of the number of schema objects as well as their names  $RDB \sim SDB$ . Both make use of names defined in the CityGML 2.0 conceptual schema. The number of tables and graphs is equal in both databases. Graph names are equivalent to table names, when underscores are removed from table names. For each table and a graph with the corresponding name, the number of columns in the table is equal to the number of edge names in the graph. There exists an equivalent graph edge name in a graph with name equivalent to a table name, for each table column name, when underscores are removed from table names and column names. The number of rows in each table is equal to number of sets of triples sharing the same subject in the graph with equivalent name. For each value of each row of each table, there exists an equivalent graph vertex, in a graph with the name equiv-



**Figure 2:** An example of structural isomorphism illustrating equivalence of the city model representations within the original 3D City Database and its semantic twin. Named graphs of addresses and buildings correspond to relational database tables with the same names. IRIs identifying entities correspond to table records with unique and sequential IDs as primary keys of the original database. Address to building binding occurs by linking data via their IRIs, instead of via records containing appropriate foreign keys in the original binding table.

alent to that table name, which is an endpoint of a graph edge with a name equivalent to a column name to which this value belongs in that table, when underscores are removed from table names and column names. In other words, the above describes morphisms  $f : RDB \rightarrow SDB$  and  $g : SDB \rightarrow RDB$ , where  $g$  is an inverse of  $f$ . The established isomorphism stays in accordance with the transformation approach described by Tim Berners Lee as one of the ways of publishing data from relational databases as linked data [7].

**Table 2:** High level overview of building blocks of a dynamic city model representation as well as their implementations within original 3D City Database and its semantic twin (from left to right).

Building Block	3D City Database	Semantic 3D City Database
Schema	SQL database schema	OntoCityGML Tbox
Features	tables	named graphs
Features' Properties	columns	OntoCityGML predicates
Instance Representation	rows	same subject triple sets
Instance Data	rows' values of each column	triples' objects
Instance IDs	sequential numbers	URIs
Instance Relations	numerical foreign keys	URI subjects
Query Language	SQL	SPARQL

The open world assumption (OWA) in the semantic representation [5] is what makes it different from the relational database representation with closed world assumption (CWA) [54]. An analogy could be drawn to the  $10 = 9.9999\dots$  equation. It is possible to say much more about the world when one has the realm of real numbers at hand to describe

it – as on the right side of the relation – than it is when one is left to do so with only decimal numbers at hand – as on the left side of the relation. The same statement holds when considering representations built on elements listed in Table 2.

The following example illustrates one of the potential consequences of CWA and OWA when considering relational versus semantic 3D City Databases used to build the proof of concept described in this paper. Both databases contain information concerning a building identified by gml id BLDG\_000300000007a403, which could be also found in the original CityGML 2.0 representation of Charlottenburg-Wilmersdorf downloaded from <https://www.businesslocationcenter.de/en/economic-atlas/download-portal/> for the purpose of building this proof of concept. Both databases also contain information about an address, identified by the gml id UUID\_76daf80a-2fef-443d-88bb-b9bc0c24fffb belonging to this building. While querying for information concerning the building and its address in both databases, it is possible to find that this building is in Berlin at 36 Taurogener Str. One can also find out that the building is bounded by a polygon specified by the following set of coordinates: {<384781.38838, 5820924.88493, 33.54>, <384789.35344, 5820924.88493, 33.54>, <384789.35344, 5820938.34387, 36.13122>, <384781.38838, 5820938.34387, 36.13122>} as well as that it has one roof specified by another polygon with a slightly different set of coordinates, in both databases.

However, the relational database address record for the building contains NULL in place of the country, whereas the semantic database contains a BLANK NODE as a vertex, which is an endpoint of the country edge connected to the address of the building on the other side. In the case of the relational database, under the CWA, this could be interpreted as “The building with this address does not belong to any country”. In case of its semantic twin and under the OWA, on the other hand, it is possible to interpret that as “It is not known to which country the building with this address belongs to”.

This would matter if both databases were integrated into a bigger system, like TWA, and turned into knowledge bases. For example, one could imagine a subsystem integrating different information sources coming from a few neighbouring European countries in order to find out the most suitable roof locations for solar panels, for instance. It is not hard to imagine that some of those information sources would not contain any geospatial information about the buildings, but would rather make possible retrieval of some information for buildings by postal address. Geospatial search, as one of the features of the dynamic geospatial knowledge graph in TWA, would allow retrieval of information concerning buildings with one roof in a square area spanning those countries and compare this information with the one retrieved from the information sources containing no geospatial information whatsoever.

It would be not possible to easily integrate those systems and make them interoperable under the CWA. On the one hand, one would end up with the information containing the one roof buildings in a few countries, specified by the coordinates of the square. On the other hand, one would end up with buildings with either no country, under the CWA, or unknown country, under the OWA. It would be easier to add the country information to such buildings by narrowing down the geospatial search to the country level after that and filling in the missing information under the OWA.

This way, it would be possible to say that “The building identified by gml id BLDG\_00030-





0000007a403 is in Germany” and integrate this information with other systems, which do not contain any geospatial information for European buildings. However, under the CWA, one would end up with two contradictory statements: “The building identified by gml id BLDG\_000300000007a403 is in Germany”, and “The building identified by gml id BLDG\_000300000007a403 does not belong to any country”. Contradictory statements contain no information and, therefore, it is possible to say much more about the world under the OWA. Adding new information onto the CWA system “The building with the address identified by the gml id BLDG\_000300000007a403 is in Germany” changes such a system and invalidates previous inferences, whereas such a thing does not take place in the OWA systems.

## 4 Semantic 3D City Data Store

Results from working on the OntoCityGML ontology and augmented data transformation tool, described in previous sections, ensured the possibility of city model representations compliant with W3C and OGC standards at the same time. In order to obtain this result, promising when looked at from the point of view of sustainable digitalisation practices, realised in a form of dynamic geospatial knowledge graph, Semantic 3D City Database had to be created within a scalable and W3C compliant triple store as well. To keep the architecture open to further collaborations and maximise potential of its reuse and innovative modifications in the future, during the proof of concept stage of the database, research presented here was focused on open source stores. From the scalability point of view, the store must be capable of accommodating city data in a form of semantic triples. Furthermore, it must be possible to add more data without significantly reducing performance of geospatial queries. In addition to that, the store must be capable of ensuring multidomain interoperability by allowing city data to be linked with any other data in a semantic form and be queried for such relationships. The following describes results of the research, briefly summarised in Figure 3, as well as motivations of the final triple store technology choice as a target solution for this proof of concept.

Considering Eclipse RDF4J, an open source framework for processing Resource Description Framework (RDF) data [19], as a triple store for the Semantic 3D City Database was motivated mainly by its relative popularity as well as familiarity. It would allow implementation of TWA data interoperability within a dynamic geospatial knowledge graph without the need for data migration. At the moment, there are a few existing TWA components which use it as a data store backend. The framework has modular architecture. It is composed of a parser/writer API, model API, repository API and a storage and inference layer (SAIL) API. Its repository API supports SPARQL 1.1 query and update language. Different core database implementations are also supported, such as memory store, native store and elastic search store. On top of these core databases, the RDF4J API can be extended with SPARQL Inferencing Notation (SPIN) rule based reasoning functionalities [37]. The RDF4J framework implements GeoSPARQL functions, but it fails almost all of the GeoSPARQL benchmark tests [33]. The SAIL interface can be successfully used for communication between the RDF4J framework and an Apache HBase database in order to process petabytes of heterogeneous RDF data [55]. However, limited geospa-



 <p><b>Pros:</b></p> <ul style="list-style-type: none"> <li>- popular, familiar</li> <li>- currently a lot of JPS data is stored in this form</li> </ul> <p><b>Cons:</b></p> <ul style="list-style-type: none"> <li>- limited/none geospatial support</li> <li>- limited scalability</li> </ul>	 <p><b>Pros:</b></p> <ul style="list-style-type: none"> <li>- rich geospatial support</li> <li>- familiar rdf4j backend</li> </ul> <p><b>Cons:</b></p> <ul style="list-style-type: none"> <li>- not documented very well</li> <li>- limited scalability of rdf4j backend</li> <li>- impression of not being production ready</li> <li>- no existing technical community forums</li> <li>- never used with JPS before</li> </ul>
 <p><b>Pros:</b></p> <ul style="list-style-type: none"> <li>- popular, familiar</li> <li>- currently a lot of JPS data is stored in this form</li> </ul> <p><b>Cons:</b></p> <ul style="list-style-type: none"> <li>- limited/none geospatial support</li> <li>- limited scalability</li> </ul>	 <p><b>Pros:</b></p> <ul style="list-style-type: none"> <li>- enterprise grade</li> <li>- geospatial support</li> <li>- tested in production by big names (ie. Wikidata)</li> <li>- very scalable</li> <li>- commercial and cloud versions available</li> </ul> <p><b>Cons:</b></p> <ul style="list-style-type: none"> <li>- never used with JPS before</li> </ul>

**Figure 3:** Triple stores taken under consideration while selecting the most suitable data storage, providing scalable geospatial search functionality for the Semantic 3D City Database. Pros and cons of each solution are listed next to their logos. Blazegraph™ was selected as the most promising one.

tial support as well as limited out of the box scalability motivated further research on the triple store of choice for the semantic geospatial database storing city models.

Another open source SPARQL server project, Apache Jena Fuseki, was also considered because of its relative popularity and familiarity (use of and interoperability with TWA components). It has been used as a triple store backend for some of TWA components as well. Users can run it as an operating system service, Java web application or a standalone server. The server follows the SPARQL 1.1 protocol to query and update RDF data [4]. It also provides a graph store protocol [56]. A Fuseki SPARQL server evaluation test shows that it is too slow to be used in the production of software for intensive use [34]. Apache Jena Fuseki supports a HTTP server component that conforms to the GeoSPARQL standard [25]. A GeoSPARQL compliance benchmark test used thirty benchmark requirements to prove that Jena Fuseki can handle geographical vector data representation literals. The Jena Fuseki server supports top level spatial and topological relation vocabulary components, as well as Resource Description Framework Schema (RDFS) entailment [33]. Because geospatial search support has been set as an essential requirement for the dynamic geospatial knowledge graph, the lack of it, as well as limited scalability, motivated further research on the triple store of choice for the Semantic 3D City Database proof of concept.

Although the spatiotemporal store Strabon, provided as an open source project by the University of Athens, has never been used with TWA or any of its components before, it caught initial attention during research on semantic data stores for the dynamic geospatial knowledge graph as well. This was partially for the reasons discussed before; because it uses familiar rdf4j backend as one of its components. Moreover, the store provides rich

geospatial support and implementation of GeoSPARQL, stSPARQL, GML and WKT literals [40]. Its architecture is also based on using named graphs to separate data. Strabon is known to show good performance on single machine and synthetic datasets [24]. The novelty of the underlying stRDF model and stSPARQL query language consists of adding temporal extensions to the semantic representations [42]. Although the mentioned query language and model has not yet made it to the realm of W3C standards, from their authors' perspective they provide a major advantage over pure GeoSPARQL. However, in the context of TWA, a time varying knowledge graph was already considered during development of its Parallel World Framework [21]. Within TWA, this approach will be explored further instead, as it was conceptualised to address a much broader problem spectrum than that. Strabon scales up to 500 million triples [42]. This is less than double the estimated number of triples required to transform the entire Berlin CityGML 2.0 data available at the moment. It would be hard to link other datasets with the city data and provide a sufficiently rich multi-domain interoperability for TWA under those limits. Moreover, apart from the limited scalability of the pure rdf4j backend, there are still a number of open problems related to scalability and inferencing with stSPARQL and its underlying data model [37]. Some of them are solved by using PostGIS in addition to rdf4j within the system. However, Semantic 3D City Database is a proof of concept demonstrating the possibility of utilising fully non relational data stores for geospatial representations, so that they can be incorporated into dynamic geospatial knowledge graphs. Strabon also appears less than production ready for larger deployments, due to its relatively rudimentary documentation [58] when compared to other RDF stores. An additional point is a lack of technical community forums which would allow users to find answers to common questions and help with resolving any potentially arising issues. Geospatial search as a feature is also not mentioned in the currently available documentation or publications concerning the store.

Blazegraph™ is an active open source project and a triple store which met the requirements listed at the beginning of this section. It is a W3C compliant semantic data store released under a GPL-2.0 License [8]. Because of its compliance with standards, it proved to be relatively easy to migrate other TWA data to this triple store as well, even if it had not been used within the system before. The latest stable version, 2.5.1, was released on the 19th of March 2019. The latest version candidate, 2.6.1, was released on the 4th of February 2020. Blazegraph™ is in production use at Fortune 500 companies such as EMC, Autodesk and Wikimedia Foundation's Wikidata Query Service. Semantic transformation of Charlottenburg-Wilmersdorf CityGML 2.0 LOD2 data contains 20,570 buildings and results in 24,244,610 triples materialised and stored in Blazegraph™ across multiple named graphs in a single namespace. Blazegraph™ supports up to 50 billion edges on a single machine and it seems to be capable of accommodating the whole Berlin city data, which is split into 12 parts (Charlottenburg-Wilmersdorf being one part). Assuming that each part contains between 20,000 and 25,000 buildings, there will be a need to accommodate between 240,000 and 300,000 buildings in order to semantically represent the whole of Berlin. Assuming uniform complexity of the buildings in other parts of the city, it would result in between 282,873,427 and 353,591,784 semantic triples generated. This is still quite far from reaching the single machine limit and shows the possibility of integrating various additional layers of heterogeneous data to complement the city data and achieve high levels of multi-domain data interoperability within a general knowledge



graph, such as TWA.

Blazegraph™ supports geospatial search via SPARQL queries. Partitioning data into namespaces allows for query optimisation by executing them on smaller portions of data, potentially in parallel. Using named graphs for different parts of city objects (i.e. walls, roofs, etc.) allows querying smaller graphs within namespaces independently. Information resulting from such independent queries could be combined into information about larger objects, as well as various interdependencies between such objects. The linked data approach allows OntoCityGML buildings' data to be combined with other semantic data, either stored within one and the same namespace, or across named graphs in separate namespaces. Federated queries are supported by Blazegraph™ too. Sale-out and High Availability features are available in Enterprise editions of Blazegraph™, which also supports GPU query optimisation, amongst many other features. Transactions, very high concurrency and very high aggregate IO rates are supported in all of its editions.

In conclusion, Blazegraph™ triple store has been used for this proof of concept because of its scalability as well as geospatial search algorithms already implemented as a part of its functionality. Enabling this functionality in Blazegraph™ required development of a custom vocabulary class as well as a datatype configuration properties file. Due to the variety of the geometrical shape types found in the proof of concept data, it was not possible to create such a configuration manually. Therefore, a functionality of automatically creating such datatype configurations as well as corresponding vocabulary items was added to the augmented TUM data transformation tool's *GeometryConverterAdapter* class. Together with a newly introduced *BlazegraphConfigBuilder* class, based on a thread safe singleton design pattern, the tool detects any new shape type not previously encountered in the data and creates appropriate configurations based on its geometrical properties. Those properties are also encoded in vocabulary item names to make it easier to break down the stored data into underlying geometries. This way, for instance, it is possible to find out that coordinates stored under the datatype ending with SOLID-3-15-15-15-15-15-15, as the last part of the IRI, contain information about a cube. The first number – 3 – describes dimensionality of the stored geometry type. The rest of the numbers say that the stored data consists of 6 parts of such a cube and each piece describes a polygon in terms of  $15/3 = 5$  points encoded in a coordinate system. This algorithm allowed detection and building of datatype configurations automatically for over 2000 different geometrical shape types found in the Charlottenburg-Wilmersdorf LOD2 building data. Those were required to fully complete and materialise the Semantic 3D City Database proof of concept, enabling the possibility of dynamic geospatial knowledge graph components within TWA. Sample live query results performed on the Charlottenburg-Wilmersburg data, with the Semantic 3D City Database already integrated into TWA, and particularly relevant to the city planning, are presented in the next section together with hardware requirements, estimates and recommendations.

## 5 Hardware Requirements

Integration of the Semantic 3D City Database, populated with Charlottenburg-Wilmersdorf LOD2 building data, by transforming the original CityGML 2.0 representation using the

**Table 3:** City planning related questions answered by the live CKG subsystem of TWA.

Query No.	Question	No. of Solutions	Elapsed Time
1.	Ask if a certain building function exists in the dataset.	1	2 ms.
2.	Ask if a certain street name exists in the dataset.	1	219 ms.
3.	Return all data about a building in a specified address.	39	193 ms.
4.	Return all distinct generic attribute names found in the dataset.	40	602 ms.
5.	Return all distinct building function codes in the dataset.	167	155 ms.
6.	Return all generic attribute names and their values found in the dataset.	956 823	22118 ms.
7.	Return specified generic attribute “Qualitaet” with all its values found in the dataset and order solutions by the value in descending order.	2 798	6048 ms.
8.	Return all distinct street names found in the dataset. Count the number of buildings in every street.	738	199 ms.
9.	Return all distinct street names that have building function code “1134” in it. Count how many times that function occurred and order streets by the number of function occurrence in descending order.	7	155 ms.
10.	Return all distinct street names that have building function code “1444” in it. Count the ratio of the specified function in each street. Order results by the ratio in descending order.	89	480 ms.

augmented Importer/Exporter tool and OntoCityGML ontology, into the TWA, enabled dynamic geospatial knowledge graph capabilities within the wider system. Cities Knowledge Graph (CKG) is a TWA subsystem under active development and research [13] in collaboration between the Cambridge Centre for Advanced Research and Education in Singapore (CARES) [12] and the Singapore-ETH Centre (SEC) [14]. A decision support system for Smart City planning, based on the CKG, is a showcase of making sustainable urbanisation practices while being aided by the sustainable digitisation practices described in the previous sections. Results of sample questions, important from the city planning perspective, translated into the appropriate SPARQL queries executed against the CKG are listed in the tables 3 and 4. The number of query solutions as well as elapsed time are listed next to each query.

The system is deployed to a server with Microsoft Windows Server 2016 Standard as an operating system and 1TB of storage space, 200GB RAM as well as 2 Intel® Xeon® ES-2620 v3 @ 2.40GHz CPUs. Out of the total 200, 32GB of RAM is assigned solely to the Blazegraph™, deployed in Nano SPARQL Server mode. The current journal file size used by it to store available city data is 6.13GB. System performance on the type of queries illustrated in Tables 3 and 4 is satisfactory in a single user mode.

Following are the results of executing multiple geospatial search queries simultaneously

**Table 4:** *City planning questions answered by the live CKG subsystem of TWA. (continued from Table 3)*

Query No.	Question	No. of Solutions	Elapsed Time
11.	Return all street names that have building function code “1171” in it. Return other existing functions in those streets. Count and order other functions by their occurrence in descending order.	66	415 ms.
12.	Return the average height of each function code found in the dataset and order functions by average height in descending order.	167	172 ms.
13	Return all distinct street names found in the dataset. Count average, minimum and maximum height of each street. Order solutions by average height a descending order."	689	219 ms.
14.	Return all distinct streets found in the dataset. Order solutions by the number of buildings in the street and by the average height in ascending order.	689	225 ms.
15.	Return all distinct street names that have at least one of the specified function codes. Order solutions by the number of matched function codes in descending order and by the average street height in ascending order.	71	338 ms.
16.	Return all addresses of buildings with function code “2921” found in the dataset. Order results by street name and number in ascending order.	55	156 ms.
17.	Return all addresses and envelope coordinates of buildings with function code “2921” found in the dataset.	40	164 ms.
18.	Return all building Ids and their function codes found within a given boundary.	93	5073 ms.
19.	Return function codes found within a given boundary. Count function code occurrence and order functions by their occurrence in descending order.	12	4938 ms.
20.	Return a list of building function codes found within the given boundary. Count the ratio of these functions in the given boundary. Order results by ratio in descending order.	6	4152 ms.

on the CKG, using the same hardware. This incremental geospatial search concurrency test follows the first few numbers of the Fibonacci Sequence, each multiplied by 10.

- 10 concurrent geospatial search queries completed in 611 ms, increasing overall system CPU utilisation by 3% and memory utilisation by 0%. Queries returned 3913 city objects contained in variable square size areas in total.
- 20 concurrent geospatial search queries completed in 1617 ms, increasing overall system CPU utilisation by 4% and memory utilisation by 0%. Queries returned 28653 city objects contained in variable square size areas in total.

- 30 concurrent geospatial search queries completed in 2771 ms, increasing overall system CPU utilisation by 8% and memory utilisation by 0%. Queries returned 52196 city objects contained in variable square size areas in total.
- 50 concurrent geospatial search queries completed in 3959 ms, increasing overall system CPU utilisation by 9% and memory utilisation by 0%. Queries returned 76689 city objects contained in variable square size areas in total.
- 80 concurrent geospatial search queries completed in 5766 ms, increasing overall system CPU utilisation by 10% and memory utilisation by 0%. Queries returned 110318 city objects contained in variable square size areas in total.
- 130 concurrent geospatial search queries completed in 9274 ms, increasing overall system CPU utilisation by 11% and memory utilisation by 0%. Queries returned 184496 city objects contained in variable square size areas in total.
- 210 concurrent geospatial search queries completed in 12292 ms, increasing overall system CPU utilisation by 11% and memory utilisation by 0%. Queries returned 255711 city objects contained in variable square size areas in total.

Geospatial search queries were sent to the `/citieskg/namespace/berlin/sparql` endpoint in the TWA as collections of HTTP GET requests, using Postman v8.1.0 [49] web API testing tool, over the Internet. The proof of concept CKG, populated with Charlottenburg-Wilmersdorf LOD2 building data and integrated into TWA, shows satisfactory results in tests of concurrent execution of geospatial search queries as well, when looked at from the point of view of expected workloads.

Recommendations for larger and more intensive workloads vary. For instance, Nguyen and Kolbe [46] test graph comparison algorithms on city data using a machine running SUSE Linux Enterprise Server 12 SP1 (64 bit) equipped with Intel® Xeon® CPU E5-2667 v3 at 3.20GHz (16 CPUs + Hyper-threading), a PCIe Solid-state Drive Array (SSD) and 1 TB of main memory. Whereas RDF GAS API, implemented in Blazegraph™, seems not to be particularly heavy on memory and CPU requirements, but instead it emphasises the importance of the SSD technology to achieve close to 1 million traversed edges per second on a MacBook Air [10]. Upon consultation, one of the market-leading server hardware vendors recommended the following configuration for the CKG – at least, as a starting point to a system able to store and query between 282,873,427 and 353,591,784 semantic triples generated in case of the whole Berlin and integrated within TWA:

1. Frontend web server:

- 12 core / 2.9GHz CPU
- 64GB RAM
- 2x 300GB HDD

2. Virtualisation server for more intensive operations on geospatial data:

- 32 core / 2.3GHz CPU
  - 256GB RAM
  - 2 x 300GB HDD
3. Expandable Network Attached Storage system for VMs:
- 16TB usable, hot-swappable and expandable SSD storage.

## 6 Conclusions and future work

Applying sustainable digitisation practices in order to build scalable technological solutions, based on standards and open source components, could be used to create intelligent decision support systems. Architecture definition for one of them, in the form of CKG, elaborated on in this paper, shows that incorporation of the Semantic 3D City Database into TWA, enables dynamic geospatial knowledge graph capabilities within it and makes it able to aid sustainable urbanisation practices and decision making processes.

Previous sections demonstrate how to build knowledge graphs capable of semantic representation of three-dimensional geometrical city objects and, in this way, provide comprehensive insights based on multi-domain data interoperability. A refined OntoCityGML ontology, presented in Section 2, is an example of bringing a well-defined and specified international standard, namely CityGML 2.0, into the world of semantic web and making it compliant with W3C recommendations and standards at the same time. The demonstration of using such ontology with augmented data transformation tools proved suitability of the ontology to serve as a schema for the semantic twin of the 3D City Database, designed and optimised at TUM for many years. CKG leverages this past work by keeping data in named graphs organised within namespaces. Advantages of the new semantic representation arising from the implied OWA have been shown when adding intelligence to a bare database is of interest as well. The last sections show the possibility of materialisation of such a database within a scalable triple store with geospatial search capabilities. The store also adheres to W3C standards as well as making it possible to integrate geospatial data within a general knowledge graph, such as TWA, and provide multi domain data interoperability. Such systems could be hardware demanding. Sample tests of a live CKG single user and concurrent queries gauge existing TWA capabilities with regard to that, allowing better understanding of how to evolve this aspect of the system further. Such recommendations conclude the last section of this paper.

TWA, at its core, is an agent-based system where intelligent autonomous agents operate on the knowledge graph. A system of agents specific to the CKG has not been considered during its proof of concept as presented in this paper. In order to do that, further strengthening of the OntoCityGML would be required, mainly consisting of cross-checking more CityGML 2.0 concepts not found in the sample city data used in this proof of concept. This would allow to add more data into the CKG and enable it to serve as a base for even broader insights with such data linked to other datasets, already available in TWA. At the same time, further extensions of the TUM data transformation tools would be needed as well. Higher level geospatial search functionalities currently implemented Blazegraph™,

namely *inCircle* and *inRectangle* [9], failed in some of the tests conducted using different coordinate reference systems. Therefore CKG makes use of only *customFields* geospatial search feature at the moment. Resolving issues with those higher level types of search would add more capabilities and enable implementation of certain functionalities going forward as well. Making use of the Parallel World Framework, already implemented in TWA, with the city data would allow users to simulate, analyse, dynamically visualise and evaluate various urbanisation scenarios and enable the possibility for cross domain sustainability impact analyses.

One can not forget that TWA, as an information system, also occupies some physical space in a data centre – a group of buildings used to house computer systems and their associated components. Improving the sustainability of such buildings has come into focus for many big technology companies in recent years. Very often they look into making use of more efficient cooling as well as utilisation of renewable energy sources, such as installation of solar panels on their roofs, as well as multi-tenancy or maximisation of optimal computing resource utilisation, eliminating idle but energy-consuming times. When looked at from this angle, the presented proof of concept may be regarded as a brick on paving the road towards self-sustainable knowledge graphs.

## Acknowledgements

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme. Markus Kraft gratefully acknowledges the support of the Alexander von Humboldt foundation.

The research was conducted as part of an Intra-CREATE collaborative project involving CARES (Cambridge Centre for Advanced Research and Education in Singapore), which is University of Cambridge's presence in Singapore, and Future Cities Laboratory at the Singapore-ETH Centre, which was established collaboratively between ETH Zurich and the National Research Foundation Singapore.

## List of abbreviations

<b>EIP</b>	<b>Eco-Industrial Park</b>
<b>HTTP</b>	<b>Hypertext Transfer Protocol</b>
<b>URL</b>	<b>Uniform Resource Locator</b>
<b>URI</b>	<b>Uniform Resource Identifier</b>
<b>IRI</b>	<b>Internationalized Resource Identifier</b>
<b>JPS</b>	<b>J-Park Simulator</b>
<b>JSON</b>	<b>JavaScript Object Notation</b>
<b>OWL</b>	<b>Web Ontology Language</b>
<b>RDF</b>	<b>Resource Description Framework</b>
<b>TUM</b>	<b>Technische Universität München</b>
<b>SQL</b>	<b>Structured Query Language</b>
<b>SPARQL</b>	<b>SPARQL Protocol and RDF Query Language</b>
<b>W3C</b>	<b>World Wide Web Consortium</b>
<b>XML</b>	<b>Extensible Markup Language</b>
<b>GML</b>	<b>Geography Markup Language</b>
<b>JDBC</b>	<b>Java Database Connectivity</b>
<b>TWA</b>	<b>The World Avatar</b>
<b>OWA</b>	<b>Open World Assumption</b>
<b>CWA</b>	<b>Closed World Assumption</b>
<b>DL</b>	<b>Description Logic</b>
<b>OBDA</b>	<b>Ontology-Based Data Access</b>
<b>OGC</b>	<b>Open Geospatial Consortium</b>
<b>WKT</b>	<b>Well-Known Text</b>
<b>CKG</b>	<b>Cities Knowledge Graph</b>
<b>CARES</b>	<b>Cambridge Centre for Advanced Research and Education in Singapore</b>
<b>SEC</b>	<b>Singapore-ETH Centre</b>
<b>LOD2</b>	<b>Level Of Detail 2</b>
<b>CIM</b>	<b>City Information Model</b>
<b>DT</b>	<b>Digital Twin</b>

## A OntoCityGML knowledge-base expressed in a DLs syntax

Concept inclusion (CI) axioms:

*CityModelType*  $\sqsubseteq$  *AbstractFeatureCollectionType*  
*CityModelType*  $\sqsubseteq$   $\forall\_GenericApplicationPropertyOfCityModel.anyType$   
*CityModelType*  $\sqsubseteq$   $\exists appearanceMember.AppearanceType$   
*CityModelType*  $\sqsubseteq$   $\forall boundedBy.EnvelopeType$   
*CityModelType*  $\sqsubseteq$  *CityGMLCoreModule*  
*CityModelType*  $\sqsubseteq$   $\forall cityObjectMember.AbstractCityObjectType$   
 $\top$   $\sqsubseteq$   $\forall cityObjectMember.AbstractFeatureType$   
*BuildingType*  $\sqsubseteq$  *AbstractBuildingType*  
*AbstractBuildingType*  $\sqsubseteq$  *BuildingModule*  
*BuildingModule*  $\sqsubseteq$  *CityGMLModule*  
*CityGMLModule*  $\sqsubseteq$  *citygml*  
*citygml*  $\sqsubseteq$   $\top$   
 $\top$   $\sqsubseteq$   $\forall Building.BuildingType$   
*BuildingType*  $\sqsubseteq$   $\forall\_GenericApplicationPropertyOfBuilding.anyType$   
 $\top$   $\sqsubseteq$   $\forall\_BoundarySurface.AbstractBoundarySurfaceType$   
 $\top$   $\sqsubseteq$   $\forall CeilingSurface.CeilingSurfaceType$   
 $\top$   $\sqsubseteq$   $\forall ClosureSurface.ClosureSurfaceType$   
 $\top$   $\sqsubseteq$   $\forall FloorSurface.FloorSurfaceType$   
 $\top$   $\sqsubseteq$   $\forall GroundSurface.GroundSurfaceType$   
 $\top$   $\sqsubseteq$   $\forall InteriorWallSurface.InteriorWallSurfaceType$   
*OuterCeilingSurfaceType*  $\sqsubseteq$  *AbstractBoundarySurfaceType*  
*AbstractBoundarySurfaceType*  $\sqsubseteq$  *BuildingModule*  
*OuterFloorSurfaceType*  $\sqsubseteq$  *AbstractBoundarySurfaceType*  
 $\top$   $\sqsubseteq$   $\forall RoofSurface.RoofSurfaceType$   
 $\top$   $\sqsubseteq$   $\forall WallSurface.WallSurfaceType$   
*MultiSurfaceType*  $\sqsubseteq$  *GeometricAggregateType*  
*GeometricAggregateType*  $\sqsubseteq$  *GeometryType*  
*MultiSurfaceType*  $\sqsubseteq$  *GeometryType*  
*GeometryType*  $\sqsubseteq$  *citygml*  
*PolygonType*  $\sqsubseteq$  *SurfaceType*



Concept inclusion (CI) axioms (continue):

$SurfaceType \sqsubseteq GeometricPrimitiveType$   
 $LinearRingType \sqsubseteq GeometryType$   
 $CompositeSurfaceType \sqsubseteq CurveType$   
 $CurveType \sqsubseteq GeometricPrimitiveType$   
 $GeometricPrimitiveType \sqsubseteq GeometryType$   
 $CompositeSurfaceType \sqsubseteq GeometricComplexType$   
 $GeometricComplexType \sqsubseteq GeometryType$   
 $GeometricComplexType \sqsubseteq TransportationProperty$   
 $TransportationProperty \sqsubseteq Property$   
 $Property \sqsubseteq \top$   
 $SolidType \sqsubseteq GeometricPrimitiveType$   
 $SolidType \sqsubseteq \exists exterior.SurfaceType$   
 $AddressType \sqsubseteq AbstractFeatureType$   
 $AbstractFeatureType \sqsubseteq citygml$   
 $AbstractFeatureType \sqsubseteq \forall name.Datatypestring$   
 $\top \sqsubseteq \forall Address.AddressType$   
 $externalReferenceType \sqsubseteq CityGMLCoreProperty$   
 $CityGMLCoreProperty \sqsubseteq Property$   
 $externalReferenceType \sqsubseteq \exists externalObject.ExternalObjectReferenceType$   
 $AbstractGenericAttributeType \sqsubseteq CityGMLCoreProperty$   
 $StringAttributeType \sqsubseteq AbstractGenericAttributeType$   
 $\top \sqsubseteq \forall stringAttribute.StringAttributeType$   
 $IntAttributeType \sqsubseteq AbstractGenericAttributeType$   
 $\top \sqsubseteq \forall intAttribute.IntAttributeType$   
 $DoubleAttributeType \sqsubseteq AbstractGenericAttributeType$   
 $\top \sqsubseteq \forall doubleAttribute.DoubleAttributeType$   
 $AbstractGenericAttributeType \sqsubseteq CityGMLCoreProperty$   
 $DateAttributeType \sqsubseteq AbstractGenericAttributeType$   
 $\top \sqsubseteq \forall dateAttribute.DateAttributeType$   
 $\top \sqsubseteq \forall BuildingPart.BuildingPartType$   
 $BuildingFunctionType \sqsubseteq BuildingProperty$   
 $BuildingProperty \sqsubseteq Property$   
 $RoofTypeType \sqsubseteq BuildingProperty$   
 $MimeTypeType \sqsubseteq AppearanceProperty$   
 $AppearanceProperty \sqsubseteq Property$   
 $\top \sqsubseteq \forall TexCoordList.TexCoordListType$   
 $WrapModeType \sqsubseteq AppearanceProperty$

Concept inclusion (CI) axioms (continue):

*ParameterizedTextureType*  $\sqsubseteq$  *TextureType*

*ParameterizedTextureType*  $\sqsubseteq \forall \_GenericApplicationPropertyOfParameterizedTexture.anyType$

*ParameterizedTextureType*  $\sqsubseteq \forall target.TextureAssociationType$

*TextureType*  $\sqsubseteq$  *AbstractSurfaceDataType*

*AbstractSurfaceDataType*  $\sqsubseteq$  *AbstractFeatureType*

*AbstractSurfaceDataType*  $\sqsubseteq$  *AppearanceModule*

*AppearanceModule*  $\sqsubseteq$  *CityGMLModule*

*AppearanceType*  $\sqsubseteq$  *AppearanceModule*

*AppearanceType*  $\sqsubseteq \forall \_GenericApplicationPropertyOfAppearance.anyType$

*AppearanceType*  $\sqsubseteq \forall surfaceDataMember.AbstractSurfaceDataType$

$\top \sqsubseteq \forall X3DMaterial.X3DMaterialType$

*X3DMaterialType*  $\sqsubseteq$  *AbstractSurfaceDataType*

*X3DMaterialType*  $\sqsubseteq \forall \_GenericApplicationPropertyOfX3DMaterial.anyType$

*X3DMaterialType*  $\sqsubseteq \forall ambientIntensity.doubleBetween0and1$

*X3DMaterialType*  $\sqsubseteq \forall diffuseColor.Color$

*X3DMaterialType*  $\sqsubseteq \forall emissiveColor.Color$

*X3DMaterialType*  $\sqsubseteq \forall shininess.doubleBetween0and1$

*X3DMaterialType*  $\sqsubseteq \forall specularColor.Color$

*X3DMaterialType*  $\sqsubseteq \forall transparency.doubleBetween0and1$

$\exists hasGeometry.\top \sqsubseteq$  *GeometryType*

$\exists hasEnvelope.\top \sqsubseteq$  *EnvelopeType*

Role inclusion (RI) axioms:

$\_FeatureCollection \sqsubseteq T \times T$   
 $CityModel \sqsubseteq \_FeatureCollection$   
 $featureMember \sqsubseteq T \times T$   
 $cityObjectMember \sqsubseteq featureMember$   
 $\_Site \sqsubseteq \_CityObject$   
 $\_AbstractBuilding \sqsubseteq \_Site$   
 $Building \sqsubseteq \_AbstractBuilding$   
 $\_Feature \sqsubseteq T \times T$   
 $\_CityObject \sqsubseteq \_Feature$   
 $\_BoundarySurface \sqsubseteq \_CityObject$   
 $CeilingSurface \sqsubseteq \_BoundarySurface$   
 $ClosureSurface \sqsubseteq \_BoundarySurface$   
 $FloorSurface \sqsubseteq \_BoundarySurface$   
 $GroundSurface \sqsubseteq \_BoundarySurface$   
 $InteriorWallSurface \sqsubseteq \_BoundarySurface$   
 $RoofSurface \sqsubseteq \_BoundarySurface$   
 $WallSurface \sqsubseteq \_BoundarySurface$   
 $interior \sqsubseteq T \times T$   
 $exterior \sqsubseteq T \times T$   
 $Address \sqsubseteq \_Feature$   
 $externalObject \sqsubseteq T \times T$   
 $externalReference \sqsubseteq T \times T$   
 $xalAddress \sqsubseteq T \times T$   
 $stringAttribute \sqsubseteq \_genericAttribute$   
 $\_genericAttribute \sqsubseteq \_GenericApplicationPropertyOfCityObject$   
 $\_GenericApplicationPropertyOfCityObject \sqsubseteq \_GenericApplicationPropertyRelation$   
 $GenericApplicationPropertyRelation \sqsubseteq T \times T$   
 $doubleAttribute \sqsubseteq \_genericAttribute$   
 $dateAttribute \sqsubseteq \_genericAttribute$   
 $intAttribute \sqsubseteq \_genericAttribute$   
 $lod2Solid \sqsubseteq lod2Relation$   
 $lod2Relation \sqsubseteq T \times T$   
 $BuildingPart \sqsubseteq \_AbstractBuilding$   
 $\_AbstractBuilding \sqsubseteq \_Site$   
 $BuildingPart \sqsubseteq \_Site$   
 $measuredHeight \sqsubseteq T \times T$   
 $roofType \sqsubseteq T \times T$

Role inclusion (RI) axioms (continue):

*consistsOfBuildingPart*  $\sqsubseteq T \times T$   
*mimeType*  $\sqsubseteq T \times T$   
*TexCoordList*  $\sqsubseteq \_TextureParameterization$   
*\\_TextureParameterization*  $\sqsubseteq \_GML$   
*\\_GML*  $\sqsubseteq T \times T$   
*specularColor*  $\sqsubseteq T \times T$   
*imageURI*  $\sqsubseteq URI$   
*borderColor*  $\sqsubseteq T \times T$   
*wrapMode*  $\sqsubseteq T \times T$   
*texWrapMode*  $\sqsubseteq wrapMode$   
*target*  $\sqsubseteq T \times T$   
*ambientIntensity*  $\sqsubseteq T \times T$   
*surfaceDataMember*  $\sqsubseteq T \times T$   
*textureCoordinates*  $\sqsubseteq T \times T$   
*Appearance*  $\sqsubseteq T \times T$   
*diffuseColor*  $\sqsubseteq T \times T$   
*emissiveColor*  $\sqsubseteq T \times T$   
*shininess*  $\sqsubseteq T \times T$   
*transparency*  $\sqsubseteq T \times T$   
*X3DMaterial*  $\sqsubseteq \_SurfaceData$

## B Sample of unit test cases - covering CityModelType term in OntoCityGML

```
<ontodebug:testCase>
  <rdf:Description>
    <ontodebug:axiom rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">CityModelType SubClassOf
      AbstractSiteType</ontodebug:axiom>
    <ontodebug:type rdf:datatype="http://www.w3.org/2001/
      XMLSchema#boolean">>false</ontodebug:type>
  </rdf:Description>
</ontodebug:testCase>
<ontodebug:testCase>
  <rdf:Description>
    <ontodebug:axiom rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">CityModelType SubClassOf
      AbstractFeatureCollectionType</ontodebug:axiom>
    <ontodebug:type rdf:datatype="http://www.w3.org/2001/
      XMLSchema#boolean">>true</ontodebug:type>
  </rdf:Description>
</ontodebug:testCase>
<ontodebug:testCase>
  <rdf:Description>
    <ontodebug:axiom rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">CityModelType SubClassOf
      CityGMLCoreModule</ontodebug:axiom>
    <ontodebug:type rdf:datatype="http://www.w3.org/2001/
      XMLSchema#boolean">>true</ontodebug:type>
  </rdf:Description>
</ontodebug:testCase>
<ontodebug:testCase>
  <rdf:Description>
    <ontodebug:axiom rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">CityModelType SubClassOf
      CityGMLModule</ontodebug:axiom>
    <ontodebug:type rdf:datatype="http://www.w3.org/2001/
      XMLSchema#boolean">>true</ontodebug:type>
  </rdf:Description>
</ontodebug:testCase>
<ontodebug:testCase>
  <rdf:Description>
    <ontodebug:axiom rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">CityModelType SubClassOf
      AbstractCityObjectType</ontodebug:axiom>
    <ontodebug:type rdf:datatype="http://www.w3.org/2001/
      XMLSchema#boolean">>false</ontodebug:type>
```

```
</rdf:Description>
</ontodebug:testCase>
<ontodebug:testCase>
  <rdf:Description>
    <ontodebug:axiom rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">CityModelType SubClassOf
      AbstractFeatureType</ontodebug:axiom>
    <ontodebug:type rdf:datatype="http://www.w3.org/2001/
      XMLSchema#boolean">true</ontodebug:type>
  </rdf:Description>
</ontodebug:testCase>
```

## References

- [1] 3D City Database Importer/Exporter, 2021. URL <https://github.com/3dcitydb/importer-exporter>. Accessed 6th April, 2021.
- [2] A. Agoub, F. Kunde, and M. Kada. Potential of graph databases in representing and enriching standardized geodata. In *Conference: Dreiländertagung der DGPF, der OVG und der SGPFAt: Bern, Switzerland*, volume 25, 2016.
- [3] J. Akroyd, S. Mosbach, A. Bhave, and M. Kraft. The National Digital Twin of the UK – a knowledge-graph approach, 2021. Submitted for publication. Preprint available at <https://como.ceb.cam.ac.uk/preprints/264/>.
- [4] Apache Jena Fuseki documentation, 2021. URL <https://jena.apache.org/documentation/fuseki2/>. Accessed 26th March, 2021.
- [5] F. Baader, I. Horrocks, C. Lutz, and U. Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017. doi:10.1017/9781139025355.
- [6] T. Berners-Lee. Relational databases on the semantic web., 1998. URL <https://www.w3.org/DesignIssues/RDB-RDF.html>. Accessed March 22, 2021.
- [7] T. Berners-Lee. Linked data, 2006. URL <https://www.w3.org/DesignIssues/LinkedData.html>. Accessed March 22, 2021.
- [8] Blazegraph - About, 2021. URL [https://github.com/blazegraph/database/wiki/About\\_Blazegraph](https://github.com/blazegraph/database/wiki/About_Blazegraph). Accessed 20th April, 2021.
- [9] Blazegraph - GeoSpatial, 2021. URL <https://github.com/blazegraph/database/wiki/GeoSpatial>. Accessed 20th April, 2021.
- [10] Blazegraph - RDF GAS API, 2011. URL [https://github.com/blazegraph/database/wiki/RDF\\_GAS\\_API](https://github.com/blazegraph/database/wiki/RDF_GAS_API). Accessed 16th April, 2021.
- [11] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.
- [12] Cities Knowledge Graph - CARES, 2021. URL <https://www.cares.cam.ac.uk/research/cities/>. Accessed 15th April, 2021.
- [13] Cities Knowledge Graph - ResearchGate, 2021. URL <https://www.researchgate.net/project/Cities-Knowledge-Graph>. Accessed 15th April, 2021.
- [14] Cities Knowledge Graph - SEC, 2021. URL <https://fcl.ethz.ch/research/research-projects/cities-knowledge-graph.html>. Accessed 15th April, 2021.

- [15] CityGML ontology (2.0), 2021. URL <http://cui.unige.ch/isi/onto/citygml2.0.owl>. Accessed 12th March, 2021.
- [16] citygml4j Java class library and API, 2021. URL <https://github.com/citygml4j/citygml4j>. Accessed 6th April, 2021.
- [17] A. Devanand, M. Kraft, and I. A. Karimi. Optimal site selection for modular nuclear power plants. *Comput. Chem. Eng.*, 125:339–350, 2019. doi:10.1016/j.compchemeng.2019.03.024.
- [18] A. Devanand, G. Karmakar, N. Krdzavac, R. Rigo-Mariani, E. Y. S. Foo, I. A. Karimi, and M. Kraft. OntoPowSys: A power system ontology for cross domain interactions in an eco industrial park. *Energy and AI*, 1:100008, 2020. doi:10.1016/j.egyai.2020.100008.
- [19] Eclipse Foundation. RDF4J (3.6.0), 2021. URL <https://rdf4j.org/about/>. Accessed 11th March, 2021.
- [20] A. Eibeck, M. Q. Lim, and M. Kraft. J-Park Simulator: An ontology-based platform for cross-domain scenarios in process industry. *Comput. Chem. Eng.*, 131:106586, 2019. doi:10.1016/j.compchemeng.2019.106586.
- [21] A. Eibeck, A. Chadzynski, M. Q. Lim, L. K. Aditya, L. Ong, A. Devanand, G. Karmakar, S. Mosbach, R. Lau, I. A. Karimi, E. Y. S. Foo, and M. Kraft. A parallel world framework for scenario analysis in knowledge graphs. *Data-Centric Engineering*, 1:e6, 2020. doi:10.1017/dce.2020.6.
- [22] F. Farazi, J. Akroyd, S. Mosbach, P. Buerger, D. Nurkowski, M. Salamanca, and M. Kraft. OntoKin: An ontology for chemical kinetic reaction mechanisms. *J. Chem. Inf. Model.*, 60(1):108–120, 2020. doi:10.1021/acs.jcim.9b00960.
- [23] F. Farazi, M. Salamanca, S. Mosbach, J. Akroyd, A. Eibeck, L. K. Aditya, A. Chadzynski, K. Pan, X. Zhou, S. Zhang, M. Q. Lim, and M. Kraft. Knowledge graph approach to combustion chemistry and interoperability. *ACS Omega*, 5(29):18342–18348, 2020. doi:10.1021/acsomega.0c02055.
- [24] G. Garbis, K. Kyzirakos, and M. Koubarakis. Geographica: A benchmark for geospatial RDF stores (long version). In H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. Noy, C. Welty, and K. Janowicz, editors, *The Semantic Web – ISWC 2013*, pages 343–359, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-41338-4.
- [25] GeoSPARQL Fuseki documentation, W3C Recommendation, 2021. URL <https://jena.apache.org/documentation/geosparql/geosparql-fuseki>. Accessed 26th March, 2021.
- [26] J. Gil. City Information Modelling: A Conceptual Framework for Research and Practice in Digital Urban Planning. *Built Environment*, 46(4):501–527, Dec. 2020. doi:10/ghqp88.



- [27] B. Glimm, I. Horrocks, B. Motik, G. Stoilos, and Z. Wang. Hermit: An OWL 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014. doi:10.1007/s10817-014-9305-1.
- [28] G. Gröger, T. H. Kolbe, C. Nagel, and K. H. Häfele. OGC city geography markup language (CityGML) en-coding standard, 2012. URL <https://www.ogc.org/standards/citygml>. Accessed 11th March, 2021.
- [29] H. Hlomani and D. Stacey. Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey. *Sem. Web J.*, 1:1–11, 2014. URL <http://www.semantic-web-journal.net/content/approaches-methods-metrics-measures-and-subjectivity-ontology-evaluation-survey>. Accessed February 7th, 2019.
- [30] I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible SROIQ. In *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006. URL <http://www.cs.ox.ac.uk/ian.horrocks/Publications/download/2006/HoKS06a.pdf>. Accessed October 26th, 2018.
- [31] O. Inderwildi, C. Zhang, X. Wang, and M. Kraft. The impact of intelligent cyber-physical systems on the decarbonization of energy. *Energy Environ. Sci.*, 13:744–771, 2020. doi:10.1039/C9EE01919G.
- [32] Jena JDBC - A SPARQL over JDBC driver framework, 2021. URL <https://jena.apache.org/documentation/jdbc/>. Accessed 7th April, 2021.
- [33] M. Jovanovik, T. Homburg, and M. Spasic. A GeoSPARQL compliance benchmark, 2021. URL <https://arxiv.org/pdf/2102.06139.pdf>. Accessed 12th March, 2021.
- [34] V. Kilintzis, N. Beredimas, and I. Chouvarda. Evaluation of the performance of open-source RDBMS and triplestores for storing medical data over a web service. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4499–4502, 2014. doi:10.1109/EMBC.2014.6944623.
- [35] M. J. Kleinlanghorst, L. Zhou, J. Sikorski, E. Y. S. Foo, K. Aditya, S. Mosbach, I. Karimi, R. Lau, and M. Kraft. J-Park Simulator: Roadmap to smart eco-industrial parks. In *Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing*, 2017. doi:10.1145/3018896.3025155.
- [36] Knowledge Engineering @ CUI. Ontologies by KE@ISS, 2021.
- [37] H. Knublauch, J. A. Hendler, and K. Idehen. SPIN-overview and motivation. *W3C Member Submission*, 22, 2011. URL [www.w3.org/Submission/spin-overview/](http://www.w3.org/Submission/spin-overview/).
- [38] D. Kolas and T. Self. Spatially-augmented knowledgebase. In K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web*,

pages 792–801, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-76298-0. doi:10.1007/978-3-540-76298-0\_57.

- [39] R. Kontchakov, M. Rodríguez-Muro, and M. Zakharyashev. *Ontology-Based Data Access with Databases: A Short Course*, pages 194–229. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-39784-4. doi:10.1007/978-3-642-39784-4\_5.
- [40] M. Koubarakis, M. Karpathiotakis, K. Kyzirakos, C. Nikolaou, and M. Sioutis. *Data Models and Query Languages for Linked Geospatial Data*, pages 290–328. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-33158-9. doi:10.1007/978-3-642-33158-9\_8.
- [41] N. Krdzavac, S. Mosbach, D. Nurkowski, P. Buerger, J. Akroyd, J. Martin, A. Menon, and M. Kraft. An ontology and semantic web service for quantum chemistry calculations. *J. Chem. Inf. Model.*, 59(7):3154–3165, 2019. doi:10.1021/acs.jcim.9b00227.
- [42] K. Kyzirakos, M. Karpathiotakis, and M. Koubarakis. Strabon: A semantic geospatial DBMS. In P. Cudré-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. X. Parreira, J. Hendler, G. Schreiber, A. Bernstein, and E. Blomqvist, editors, *The Semantic Web – ISWC 2012*, pages 295–311, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-35176-1.
- [43] Manchester 3D Data Viewer, 2021. URL <https://manchester.virtualcitymap.de/#/>. Accessed 8th April, 2021.
- [44] B. Mazur. *When is One Thing Equal to Some Other Thing?*, page 221–242. Mathematical Association of America, 2008. doi:10.5948/UPO9781614445050.015.
- [45] M. Musen. The Protégé project: A look back and a look forward. *AI Matters*, 1(4), 2015. doi:10.1145/2757001.2757003.
- [46] S. H. Nguyen and T. H. Kolbe. A multi-perspective approach to interpreting spatio-semantic changes of large 3D city models in CityGML using a graph database. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VI-4/W1-2020:143–150, 2020. doi:10.5194/isprs-annals-VI-4-W1-2020-143-2020.
- [47] M. Pan, J. Sikorski, C. A. Kastner, J. Akroyd, S. Mosbach, R. Lau, and M. Kraft. Applying Industry 4.0 to the Jurong Island eco-industrial park. *Energy Procedia*, 75:1536–1541, 2015. doi:10.1016/j.egypro.2015.07.313.
- [48] M. Pan, J. Sikorski, J. Akroyd, S. Mosbach, R. Lau, and M. Kraft. Design technologies for eco-industrial parks: From unit operations to processes, plants and industrial networks. *Applied Energy*, 175:305–323, 2016. doi:10.1016/j.apenergy.2016.05.019.
- [49] Postman - The Collaboration Platform for API Development, 2021. URL <https://www.postman.com/>. Accessed 16th April, 2021.

- [50] W. J. Radermacher. *Official Statistics 4.0: The Era of Digitisation and Globalisation*, pages 119–156. Springer International Publishing, Cham, 2020. ISBN 978-3-030-31492-7. doi:10.1007/978-3-030-31492-7\_4.
- [51] J. Rashmi, J. M. Evrim, and A. Albizri. An open system architecture framework for interoperability (OSAFI). *International Journal of Business Information Systems*, 2020.
- [52] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 3 edition, 2010.
- [53] G. Schrotter and C. Hürzeler. The Digital Twin of the City of Zurich for Urban Planning. *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88(1):99–112, Feb. 2020. doi:10/ggvkr3.
- [54] M. Sir, Z. Bradac, and P. Fiedler. Ontology versus database. *IFAC-PapersOnLine*, 48(4):220–225, 2015. 13th IFAC and IEEE Conference on Programmable Devices and Embedded Systems.
- [55] A. Sotona and S. Negru. How to feed Apache HBase with petabytes of RDF data: An extremely scalable RDF store based on Eclipse RDF4J. In T. Kawamura and H. Paulheim, editors, *Proceedings of the ISWC 2016 Posters & Demonstrations Track*, volume 1690 of *CEUR Workshop Proceedings*, 2016. URL <http://ceur-ws.org/Vol-1690/paper35.pdf>.
- [56] SPARQL 1.1 Graph Store HTTP Protocol, 2021. URL <https://www.w3.org/TR/sparql11-http-rdf-update/>. Accessed 26th March, 2021.
- [57] A. Stadler, C. Nagel, G. König, and T. H. Kolbe. *Making Interoperability Persistent: A 3D Geo Database Based on CityGML*, pages 175–192. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-87395-2. doi:10.1007/978-3-540-87395-2\_11.
- [58] Strabon, the spatiotemporal RDF store., 2021. URL <http://www.strabon.di.uoa.gr/download.html>. Accessed 13th April, 2021.
- [59] A. von Richthofen, editor. *Urban Elements: Advanced Studies in Urban Design*. ETH Zurich, 2018. doi:10.3929/ETHZ-B-000270354.
- [60] U. Winkelhake. *Roadmap for Sustainable Digitisation*, pages 127–178. Springer International Publishing, Cham, 2018. ISBN 978-3-319-71610-7. doi:10.1007/978-3-319-71610-7\_6.
- [61] Z. Yao, C. Nagel, F. Kunde, G. Hudra, P. Willkomm, A. Donaubaue, T. Adolphi, and T. H. Kolbe. 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):5, May 2018. doi:10.1186/s40965-018-0046-7.
- [62] C. Zhang, A. Romagnoli, L. Zhou, and M. Kraft. Knowledge management of eco-industrial park for efficient energy utilization through ontology-based approach. *Applied Energy*, 204:1412–1421, 2017. doi:10.1016/j.apenergy.2017.03.130.

- [63] L. Zhou, M. Pan, J. J. Sikorski, S. Garud, L. K. Aditya, M. J. Kleinlanghorst, I. A. Karimi, and M. Kraft. Towards an ontological infrastructure for chemical process simulation and optimization in the context of eco-industrial parks. *Applied Energy*, 204:1284–1298, 2017. doi:j.apenergy.2017.05.002.
- [64] L. Zhou, C. Zhang, I. A. Karimi, and M. Kraft. An ontology framework towards decentralized information management for eco-industrial parks. *Comput. Chem. Eng.*, 118:49–63, 2018. doi:10.1016/j.compchemeng.2018.07.010.
- [65] X. Zhou, A. Eibeck, M. Q. Lim, N. Krdzavac, and M. Kraft. An agent composition framework for the J-Park Simulator – a knowledge graph for the process industry. *Comput. Chem. Eng.*, 130:106577, 2019. doi:10.1016/j.compchemeng.2019.106577.
- [66] X. Zhou, M. Q. Lim, and M. Kraft. A smart contract-based agent marketplace for the J-Park Simulator – a knowledge graph for the process industry. *Comput. Chem. Eng.*, 139:106896, 2020. doi:10.1016/j.compchemeng.2020.106896.