”CITYJSON2RDF” A CONVERTER FOR PRODUCING 3D CITY KNOWLEDGE GRAPHS

A. T. Akın[1], Ç. Cömert[1]

[1] KTU, Engineering Faculty, 61080 Trabzon, Turkey - (alpertunga, ccomert)@ktu.edu.tr

KEY WORDS: Linked Data, Knowledge Graph, Semantic Web, RDF, CityJSON, CityGML

ABSTRACT:

The increasing prevalence of 3D city models (3DCMs) in various applications, such as mixed reality and navigation, highlights the need for efficient data exchange. CityGML serves as a standard model for this purpose, encompassing geometric and semantic information in 3DCM data. To enhance interoperability, a compatible transfer mechanism is essential. This study introduces a conversion tool that transforms CityGML data into RDF, a knowledge graph (KG) format. Utilizing semantic web technologies, this conversion ensures the data's seamless integration across applications. The RDF model facilitates linking to open ontologies, promoting data circulation without loss. The tool employs CityJSON encoding for its mappable JSON structure, enabling straightforward conversion to RDF using Python components. While existing XML to RDF tools exist, this tool distinguishes itself by addressing accessibility and user intervention challenges. Linking is established by matching subject classes with relevant ontology definitions, a process dependent on developers' understanding of the CityGML data model. The tool, accessible through URL-2, is still in development, offering a promising solution for achieving effective 3DCM data interoperability.

1. INTRODUCTION

Today, 3 Dimensional (3D) data, especially city models, has an important role in our lives as a result of its use in studies such as mixed reality applications, infrastructure cadastre, solar potential analysis, navigation and robotics. As advancements in computer graphics and increases in computing unit capacities continue to progress, the use of 3DCMs (3D City Models) is becoming increasingly widespread. Since the use of 3DCM data in different applications requires the transfer of data between applications and platforms, CityGML was introduced as a data exchange format and a standard data model. Furthermore, the management and analysis of semantic 3D city models based on CityGML are essential for complex GIS simulation tasks (Yao et al., 2018).

In the CityGML model, the 3DCM data consists of geometries and semantic information. Semantic information refers to the thematic object classes, class hierarchies and attributions. Both geometries and semantic information could differentiate with respect to different application schemas and levels of details (LOD) for reusing the same data in various use cases (Gröger et al., 2008). The reusing requires predefined mappings between application schemas to prevent any loss of information, especially in semantics. Although these mappings are widely in use, there is a need for a reliably compatible transfer mechanism among applications for situations that include unforeseen, or undefined schemas (van den Brink et al., 2014). The compatible transfer is possible using semantic web technologies and the linked-data approach (Vinasco-Alvarez et al., 2020). The key idea behind is to convert the 3DCM data into a knowledge graph (KG) which carries its own definitions for each node and edge by linking to the relevant open ontologies. The nodes refer to the object classes and class individuals, and the edges refer to the relations between classes or individuals and the attribution ownership of the individuals. This conversion ensures that the data can be linked to any application and circulate live on the web without any loss of information, concisely, ensures the interoperability of the applications. Accomplishing this KG conversion uses the RDF (Resouce Description Framework) standard data model, which is the semantic web's standard data model and exchange format.

RDFs depend on the creation of triples in the form of (subject, predicate, object), where subjects and objects denote entities, and predicates establish relationships between entities (Zhang et al., 2021; Kostovska et al., 2022). RDF Schema (RDFS) extends the RDF data model and provides essential elements for describing ontologies, including classes and properties, thereby facilitating the organization and representation of knowledge in a structured manner. Moreover, RDF serves as a fundamental format for data integration and database interoperability, accelerating the integration of diverse datasets and enabling high-quality data representation (Liu et al., 2023). KGs are built upon RDF and have become essential in organizing and storing structured knowledge, playing a crucial role in various knowledge-driven applications such as question answering, sentiment analysis, and image captioning. KGs provide essential relational information between entities, nodes, contributing to the comprehensive representation of knowledge and facilitating data retrieval and inference (Lv et al., 2018). Therefore, RDF and KGs are foundational components in the representation, integration, and utilization of structured data, playing a pivotal role in advancing knowledge organization and retrieval in diverse domains. For a more deep dive into the structure of the RDF and the fundamentals of the semantic web and KGs, cited resources could be beneficial (Hitzler et al., 2009; Hogan et al., 2020). Besides, semantic web technologies make it possible to infer new ones by using the actual relationships contained in data (Vinasco-Alvarez et al., 2021). In this way, it can be possible to query implicit relationships that are not directly declared in the data model through the RDF graph.

In this study, a conversion tool for converting the 3DCM data in the CityGML data model to RDF is introduced. A literature search can bring up some “XML to RDF” or “GML to RDF” converting tools, however many of these are inaccessible (Bulen et al., 2011; van den Brink et al., 2014; Vinasco-Alvarez et al., 2020) or they do not provide linking with no end-user intervention (URL-1). The main struggle in the development of a converter is to provide the links between definitions, ontologies, and RDF components. While the most complete tool is GeoTriples for conversion of spatial data to RDF, it uses GeoSPARQL based mapping rules. If the user needs a different ontology, such as CityGML in our study, the user should edit the generated mapping document (Kyzirakos et al., 2018). This necessity could potentially pose a barrier to adoption for users with limited expertise in semantic web or ontologies. While the linking with no user intervention is still an unconcluded research topic, in our tool we also have tried to derive links using the CityGML and GeoSPARQL ontologies.

A 3DCM file in the CityJSON encoding is used for the conversion. The main reason for choosing CityJSON is the directly mappable structure of the JSON format to the dictionaries in Python language. CityJSON has also a less verbose and less nested structure compared to the GML encoding of the same data model (Ledoux et al., 2019). After the mapping to a dictionary, we composed the functions for creating RDF triples from the key-value pairs. In these functions, the linking is settled by a string matching between the subject classes and the relevant ontology definition. It should be noted that this matching is not done by an automated mechanism, it depends on the developer's, our's, knowledge of the CityGML data model. The tool can be accessed through URL-2 and is still in development.

2. METHOD

The followed pipeline of converter development is illustrated in Figure 1. The first stage is parsing CityJSON files and is handled using the "cjio (CityJSON IO)" Python library. The library provides optionalities for manipulating CityJSON files such as import, export, converting to DataFrames or converting to GLTF, validation against schemas, and etc (URL-3). Using cjio, CityJSON files are converted into DataFrame at this stage. A DataFrame is a 2-dimensional data structure introduced by Pandas Python library which can be considered as a spreadsheet or an SQL table (McKinney, 2015). A DataFrame could be constructed as a dictionary of multiple dictionaries. Converting CityJSON files to DataFrames easens to extracting triples, the main element of RDFs (Figure 2), for creating a RDF file thanks to its dictionary-like nature which includes key-value pairs. To extract triples from DataFrames, "RdfPandas" Python library is used at the next stage (URL-1). The RdfPandas library consists of two main functions, the first one provides DataFrame to RDF Graph conversion and the other one provides RDF Graph to DataFrame. In DataFrame to RDF conversion, the row indices of the DataFrame are used as subjects and the column indices as predicates, and the corresponding values of these indices become objects. The triple derivation, or RDF creation, at this stage can be seen in Figure 3. The created RDF file can be handled as a "RDFLib " object. The RDFLib is a Python library for manipulating RDF graphs (Krech, 2006).

In this state, the obtained RDFLib object is RDF-like a property graph (Hartig, 2014) without any resource linking even if it is in a known RDF file format (Figure 4). To create a complete knowledge graph (KG) from this graph, there is a need to link graph elements to the resources including relevant definitions. Considering the example above, to create a KG node from this

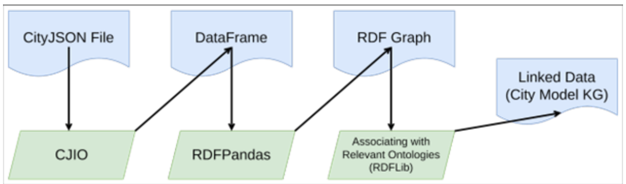

Figure 1. The followed development pipeline of the converter.

Figure 2. A triple, the main element of a RDF graph. Subject, Object and the Predicate that defines the relation between them.

triple, the definitions of "roofType", "1000" and the ID as the subject must be linked to these elements. To handle this process, CityGML (Chadzynski et al., 2021) and GeoSPARQL ontologies (Battle and Kolas, 2011) are adopted at this stage. The definitions of relations, thematic object classes, class hierarchies and attributions which belong to CityGML data model are obtained from the relevant ontology then these definitions are linked to the graph elements through URIs (Uniform Resource Identifier). In the same way, the definitions of geometry types such as point, line or polygon and spatial relations between individuals such as overlap, intersect, within are obtained from GeoSPARQL ontology. This linking operation has been performed by the developer and matching between the definitions and the elements has been done thanks to the knowledge of the developer, not in an automated way. While automated matching studies are still at an experimental level, this automation is also in our future plans. In Figure 4, the transition from RDFLib object to linked KG is illustrated.

3. RESULT

For testing the converter with real-world data, the "Den Haag.city.json" from URL-5 has been used (Figure 5). This CityJSON data has 2498 city objects including LOD2 buildings and a terrain model, and is 2.7 MB in size, while its GML encoding is 30 MB in size. The end-to-end process takes 10.7 seconds. The system configuration of the test PC is given in Table 1.

4. DISCUSSION AND CONCLUSION

The execution time of the entire pipeline must be long for a real-time web based usage. This situation occurs due to the nested nature of the data model and the notable drawback of the adopted cjio library during CityJSON to DataFrame conversion stage. This drawback is cjio's transferring only attributes to the output DataFrame, not geometries and thematic classes. To handle this drawback, additional loops and IF statements were

Table 1. The system configuration that used for development and test stages

CPU	Intel Xeon W-1250 (6 Core) (3.30GHz, 4.70GHz, 12MB)
RAM	16GB DDR4 2933MHz ECC (2x8GB)

Index	roofType	RelativeEavesHeight	RelativeRidgeHeight	AbsoluteEavesHeight	AbsoluteRidgeHeight	objectTypes	geometryTypes	parents
GUID_DBDABF53-7DD5-4C2F-BE7F-51F29A0CBA16_1	1000	3.133	3.133	7.717	7.717	BuildingPart	lod2Solid	['GUID_DBDABF53-7DD5-4C2F-BE7F-51F29A0CBA16']
GUID_8CE54418-E2F7-49A7-9A8D-C3D172BA62C4_2	1000	6.392	6.392	10.075	10.075	BuildingPart	lod2Solid	['GUID_8CE54418-E2F7-49A7-9A8D-C3D172BA62C4']
GUID_13974D93-CB4F-4B5A-AB1E-577DD9928CF2_3	1000	2.612	2.612	8.774	8.774	BuildingPart	lod2Solid	['GUID_13974D93-CB4F-4B5A-AB1E-577DD9928CF2']
GUID_8CE54418-E2F7-49A7-9A8D-C3D172BA62C4_1	1000	9.419	9.419	12.871	12.871	BuildingPart	lod2Solid	['GUID_8CE54418-E2F7-49A7-9A8D-C3D172BA62C4']
GUID_47061B3E-C490-4882-B06A-EC084006045F	nan	nan	nan	nan	nan	Building	None	CityModel
GUID_13974D93-CB4F-4B5A-AB1E-577DD9928CF2_2	1000	5.599	5.599	11.664	11.664	BuildingPart	lod2Solid	['GUID_13974D93-CB4F-4B5A-AB1E-577DD9928CF2']
GUID_13974D93-CB4F-4B5A-AB1E-577DD9928CF2_1	1130	6.108	8.711	11.864	14.467	BuildingPart	lod2Solid	['GUID_13974D93-CB4F-4B5A-AB1E-577DD9928CF2']
GUID_3D7D60B9-8F3A-4D3B-A3E5-CD9B5565A5B2	1030	6.148	9.508	11.38	14.739	Building	None	CityModel
GUID_F784906B-FD5A-4F81-BACF-79C3566B2789	nan	nan	nan	nan	nan	Building	None	CityModel
GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0_4	1010	6.991	8.415	11.27	12.693	BuildingPart	lod2Solid	['GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0']
GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0_3	1000	8.446	8.446	12.693	12.693	BuildingPart	lod2Solid	['GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0']
GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0_2	1010	6.971	8.394	11.27	12.693	BuildingPart	lod2Solid	['GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0']
GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0_1	1000	5.546	5.546	9.793	9.793	BuildingPart	lod2Solid	['GUID_1280E44F-7EFD-4BEA-8E79-3124934BD6A0']
GUID_DCAE0ACB-5F38-4AEA-80D2-7242B193AE70	nan	nan	nan	nan	nan	Building	None	CityModel
GUID_E1BD416F-8047-491A-86D0-037D2904CB2B	nan	nan	nan	nan	nan	Building	None	CityModel

Subject: "GUID_8CE54418-E2F7-49A7-9A8D-C3D172BA62C4_1",
Predicate: roofType
Object: 1000

Derived Triple: "GUID_8CE54418-E2F7-49A7-9A8D-C3D172BA62C4_1" roofType 1000

Figure 3. A triple derivation sample using RdfPandas. The highlighted row indice on the left column is the subject, the column indice roofType is the predicate and the value 1000 is the object.

```
[
  {
    "@id": "http://example.org/#id-1",
    "@type": [
      "http://www.theworldavatar.com/ontology/ontocitygml/OntoCityGML.owl#GenericCityObject"
    ],
    "http://www.opengis.net/ont/geosparql#hasGeometry": [
      {
        "@id": "http://example.org/#id-1poly"
      }
    ]
  },
  {
    "@id": "http://example.org/#id-1poly",
    "@type": [
      "http://www.theworldavatar.com/ontology/ontocitygml/OntoCityGML.owl#lod1Solid"
    ],
    "http://www.opengis.net/ont/geosparql#asWKT": [
      {
        "@type": "http://www.opengis.net/ont/geosparql#wktLiteral",
        "@value": "['POLYGON Z ((0 0 1000, 1000 0 1000, 1000 1000 1000, 0 1000 1000, 0 0 1000))',
'POLYGON Z ((0 0 0, 1000 0 0, 1000 0 1000, 0 0 1000, 0 0 0))', 'POLYGON Z ((1000 0 0, 1000 1000 0, 1000
1000 1000, 1000 0 1000, 1000 0 0))', 'POLYGON Z ((1000 1000 0, 0 1000 0, 0 1000 1000, 1000 1000 1000,
1000 1000 0))', 'POLYGON Z ((0 1000 0, 0 0 0, 0 0 1000, 0 1000 1000, 0 1000 0))', 'POLYGON Z ((0 1000 0,
1000 1000 0, 1000 0 0, 0 0 0, 0 1000 0))']"
      }
    ]
  },
  {
    "@id": "http://example.org/#model",
    "@type": [
      "http://www.theworldavatar.com/ontology/ontocitygml/OntoCityGML.owl#CityModel"
    ],
    "http://www.theworldavatar.com/ontology/ontocitygml/OntoCityGML.owl#extent": [
      {
        "@value": "POLYGON ((0 0, 0 1, 1 1, 0 0))"
      }
    ],
    "http://www.theworldavatar.com/ontology/ontocitygml/OntoCityGML.owl#transformationMatrix": [
      {
        "@value": "[0.001, 0.0, 0.0, 0.0, 0.0, 0.001, 0.0, 0.0, 0.0, 0.0, 0.001, 0.0]"
      }
    ],
    "http://www.w3.org/2002/07/owl#oneOf": [
      {
        "@id": "http://example.org/#id-1"
      }
    ],
    "http://www.w3.org/ns/odrl/2/version": [
      {
        "@value": "1.1"
      }
    ]
  }
]
```

Figure 4. The created RDF file in JSON-LD format, several highlighted examples from ontology definitions.

included to the main functions of the library for scraping the required information. The initial DataFrame output and the modified one can be seen in Figure 6. Besides this drawback, the nested nature of the data model also raised a need for additional loops and IF statements. The city objects and their surfaces could not be represented in the same DataFrame because each city object can have an imprecise count of surfaces. To overcome this, the surfaces are represented in another DataFrame with unique IDs, and IDs of the city objects occur in the second DataFrame as "parent" column index, or predicate. During the RDF graph creating stage, these two DataFrames merged by city object IDs and the parent column indexes.

Consequently, a CityJSON to RDF conversion tool for creating 3DCM KGs has been developed. Unlike other tools in the literature, we have tried to solve end-user intervention necessitate and to provide an accessible open-source tool through this link (URL-2). It should be strongly noted that this tool only eliminates the need for user intervention for purposes including the CityGML data model. Even if an automated matching between ontologies and RDF elements is still an alive research area, we have concluded it in this work by knowledge of the developer, in an hardcoded manner.

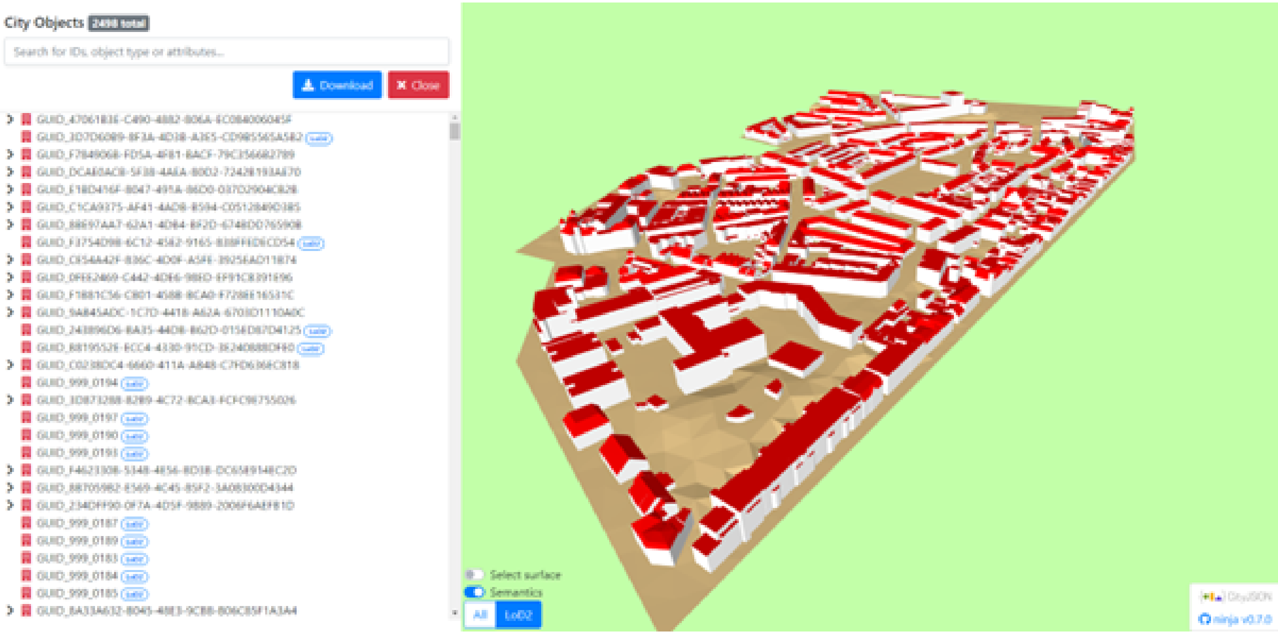

Figure 5. The used city model, “Den Haag.city.json”, visualized with ninja (URL-4).

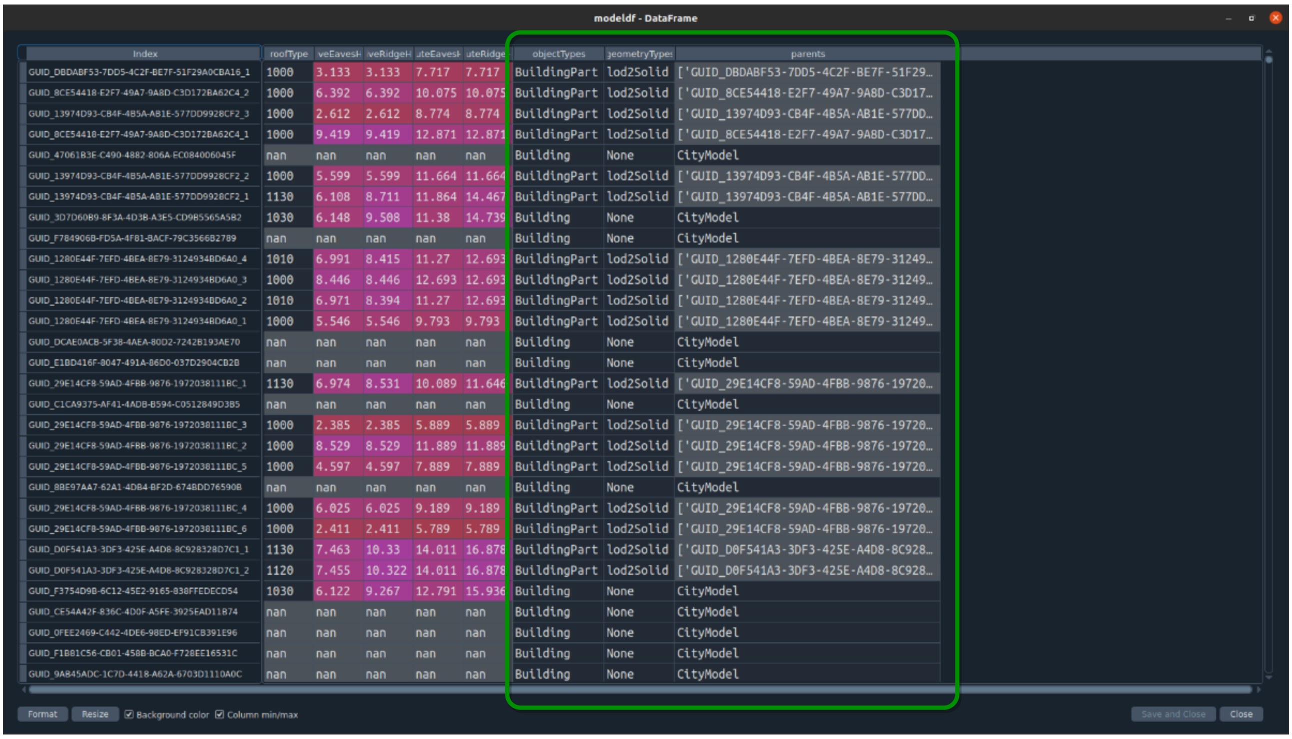

Figure 6. The highlighted with the green columns are not generated by cjio, included using additional loops and IF statements.

5. REFERENCES

Battle, R., Kolas, D. (2011). Geosparql: enabling a geospatial semantic web. Semantic Web Journal, 3(4), 355-370.

Bulen, A., Carter, J. J., Varanka, D. (2011). A program for the conversion of the national map data from proprietary format to resource description framework (rdf).

Chadzynski, A., Krdzavac, N., Farazi, F., Lim, M. Q., Li, S., Grisiute, A., ... Kraft, M. (2021). Semantic 3D City Database—An enabler for a dynamic geospatial knowledge graph. Energy and AI, 6, 100106.

Gröger, G., Kolbe, T. H., Czerwinski, A., Nagel, C. (2008). OpenGIS city geography markup language (CityGML) encoding standard, version 1.0. 0.

Hartig, O. (2014). Reconciliation of rdf* and property graphs.. https://doi.org/10.48550/arxiv.1409.3288

Hitzler, P., Krotzsch, M., Rudolph, S. (2009). Foundations of semantic web technologies. CRC press.

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., ... Zimmermann, A. (2021). Knowledge graphs. ACM Computing Surveys (Csur), 54(4), 1-37.

Kostovska, A., Vermetten, D., Doerr, B., Džeroski, S., Panov, P., Eftimov, T. (2022). Option: optimization algorithm benchmarking ontology. IEEE Transactions on Evolutionary Computation, 1-1. https://doi.org/10.1109/tevc.2022.3232844

Krech, D. (2006). Rdflib: A python library for working with rdf. Online https://github. com/RDFLib/rdflib.

Kyzirakos, K. et al. (2018). GeoTriples: Transforming geospatial data into RDF graphs using R2RML and RML mappings. Journal of web semantics, 52-53, s. 16–32. doi:10.1016/j.websem.2018.08.003

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). doi:10.1186/s40965-019-0064-0

Liu, J., Chen, J., Fan, C., Zhou, F. (2023). Joint embedding in hierarchical distance and semantic representation learning for link prediction.. https://doi.org/10.48550/arxiv.2303.15655

Lv, X., Hou, L., Li, J., Liu, Z. (2018). Differentiating concepts and instances for knowledge graph embedding. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. https://doi.org/10.18653/v1/d18-1222

McKinney, W. (2015). Pandas, python data analysis library. URL http://pandas.pydata.org, 3-15.

van den Brink, L., Janssen, P., Quak, W., Stoter, J. (2014). Linking spatial data: automated conversion of geo-information models and GML data to RDF. International Journal of Spatial Data Infrastructures Research, 9, 59-85.

Vinasco-Alvarez, D., Samuel, J. S., Servigne, S., Gesquière, G. (2020). From citygml to owl (Doctoral dissertation, LIRIS UMR 5205).

Vinasco-Alvarez, D., Samuel, J. S., Servigne, S., Gesquière, G. (2021, May). Towards a semantic web representation from a 3D geospatial urban data model. In SAGEO 2021, 16ème Conférence Internationale de la Géomatique, de l'Analyse Spatiale et des Sciences de l'Information Géographique. (pp. 227-238).

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., . . . Kolbe, T. (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). https://doi.org/10.1186/s40965-018-0046-7

Zhang, N., Jia, Q., Chen, X., Ye, H., Chen, H., Tou, H., . . . Chen, H. (2021). Alicg: fine-grained and evolvable conceptual graph construction for semantic search at alibaba. Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery Amp; Data Mining. https://doi.org/10.1145/3447548.3467057

URL-1: https://rdflib.readthedocs.io/en/stable/, Accessed: 20.12.2023

URL-2: https://github.com/alpertungakin/3DCMQuality, Accessed: 20.12.2023

URL-3: https://github.com/cityjson/cjio, Accessed: 20.12.2023

URL-4: https://ninja.cityjson.org/, Accessed: 20.12.2023

URL-5: https://www.cityjson.org/datasets/, Accessed: 20.12.2023