Development and Testing of the CityJSON Energy Extension for Space Heating Demand Calculation

Özge Tufan

First supervisor: Camilo León-Sánchez
Second supervisor: Ken Arroyo Ohori

January 25, 2022

1 Introduction

Buildings have a crucial role in the determination of environmental policies, considering that more than one-third of global energy consumption and almost 40% of energy-related CO_2 emissions are caused by buildings and construction activities (United Nations Environment Programme, 2020). To this end, the need for comprehensive knowledge about buildings and tools to process and simulate this information has boosted the importance of *Urban Energy Modelling* (Agugiaro et al., 2018), in which semantic 3D city models are frequently used by many researchers (Agugiaro, 2016b; Kaden and Kolbe, 2014; Rossknecht and Airaksinen, 2020) to obtain and store energy-related information about urban areas.

In this context, CityGML is an open data model that allows the storage and exchange of 3D city models with three official encodings: an XML-based encoding also called CityGML (Gröger et al., 2012), a JSON-based encoding called CityJSON (Ledoux et al., 2019), and as a database schema called 3DCityDB (Yao et al., 2018). CityGML allows the modelling of the most significant city objects, including buildings with their geometries as well as their semantic and thematic characteristics. Moreover, CityGML can be extended for specific use cases with the concept of Application Domain Extension (ADE) that allows the addition of classes and attributes to the data model in a systematic way.

Even though general building characteristics can be stored in CityGML, energy-related classes and attributes are not included in the data model. Therefore, Energy ADE was created to store detailed energy-related data of buildings, which can then be used in energy simulations to obtain information about the current state of buildings and for future predictions (Benner, 2018). For instance, a prominent application of Energy ADE is to assess the energy performance of buildings by focusing on specific implementations such as energy demand analysis or solar irradiation of buildings. Then, it can be used to determine future actions such as strategies on renewable energy or building refurbishment measures.

Energy ADE was provided with an XML-based encoding, and a database implementation was later created which extends the already existing 3DCityDB to store energy-related data of buildings (Agugiaro and Holcik, 2017). On the other hand, even though an Extension mechanism similar to CityGML's ADE concept is provided in CityJSON and tested with various studies ranging from the use of point clouds (Nys et al., 2021) to the addition of C-Map data structures (Vitalis et al., 2019), an energy-related CityJSON Extension is missing in the current literature.

Since the main goal of CityJSON is to create a compact and easy-to-use encoding without deep hierarchical structures and parsing problems by various software (Ledoux et al., 2019), a CityJSON Energy Extension could be the answer to bypass the complexities of Energy ADE and to store energy-related data more efficiently. The development of such an extension depends on the possibilities to reuse classes and concepts from Energy ADE while investigating the superior characteristics of CityJSON data format that may lead to higher efficiency. In addition, a use case can be chosen to test the functionalities of the Extension. Therefore, the aim of this thesis project is to *develop and test an Energy Extension for CityJSON focusing on the calculation of space heating demand of buildings*.

The use case, space heating demand of buildings, will be utilized throughout the design phase to test the functionalities and efficiency of the CityJSON Energy Extension. For this purpose, all needed input data for the use case will be calculated from the 3D city model, or obtained from external data sources to be stored in the corresponding classes or attributes in the CityJSON Energy Extension file.

This proposal is structured as follows. Following this section of Introduction, Section 2 provides an overview of the related work. Section 3 introduces the research questions as well as the scope of the thesis, and Section 4 presents the methodology that will be used in the project. Then, preliminary results are discussed in Section 5, followed by a time planning in Section 6. Finally, Section 7 provides an overview of the tools and datasets that will be used in the project.

2 Related work

This section provides a brief overview of the previous work and studies related to the thesis project. First, a short description is given for the CityGML and Energy ADE data models. Then, CityJSON encoding and its current extensions are explained. Finally, relevant studies about space heating demand calculation are introduced, with a focus on the use of 3D city models and Energy ADE.

2.1 CityGML

CityGML (version 2.0) focuses on the geometry, topology, semantics and appearance of objects that are mainly found in cities, such as buildings, bridges, vegetation and water bodies (Gröger and Plümer, 2012). The Core module of CityGML defines the basic concepts and abstract base classes, the smallest element of which is the abstract class *CityObject*. Abstract *CityObject* is the superclass of all thematic objects defined in their corresponding modules, one of which is the Building module. The Building module represents the geometry and semantics of buildings in four Levels of Detail (LoD) from LoD1 to LoD4 (Gröger et al., 2012). The base class of this module is *AbstractBuilding*, which has two subclasses called Building and BuildingPart. These classes may contain numerous properties such as the function, usage, year of construction, roof type and the number of storeys of the building.

2.2 Energy ADE

The Energy ADE (currently on version 1.0) is a data model that extends the Core and Building modules of CityGML version 2.0 to store and manage energy-related information about buildings (Agugiaro, 2016a). It is designed to be used in numerous energy applications, such as the analysis of building elements, energy demand calculations, and solar potential studies, and its scope extends from detailed single-buildings to city-scale assessments (Agugiaro et al., 2018). The Energy ADE consists of 6 thematic modules with different functionalities (Figure 1) which can be summarized as follows (Benner, 2018):

- **Energy ADE Core:** The Core module extends the *AbstractBuilding* and *CityObject* classes of CityGML with attributes and relations with new classes. Moreover, abstract base classes of the thematic modules of Energy ADE as well as new data types, enumerations, and code lists are provided.
- **Building Physics:** Thermal entities of a building (Thermal Zone, Thermal Boundary, Thermal Opening) are defined.
- **Energy Systems:** Energy conversion, distribution, storage and emission systems as well as the energy flows between them are represented.
- **Material and Construction:** Physical characteristics of construction elements and building materials are included.

- **Occupant Behaviour:** Especially used for detailed simulations, this module includes classes about a building's usage as well as its occupants and energy-related facilities.
- **Supporting Classes:** Time series, schedules and weather data are represented with various classes.

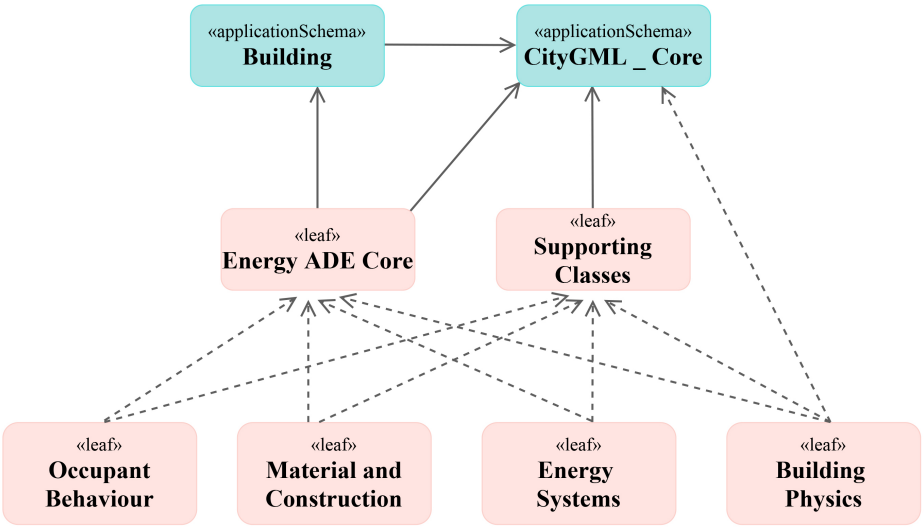

Figure 1: Modules and dependencies of Energy ADE. Adapted from Benner (2018)

Due to the complex structure of Energy ADE to represent numerous energy-related concepts, a simplified version has been created by the Karlsruhe Institute of Technology, called the KIT profile, which discards certain modules or classes from the original data model to create a subset that is easier to use (León-Sánchez et al., 2021). The main differences between the original Energy ADE and the KIT profile are that the Energy Systems module is completely removed and the Supporting Classes module is highly simplified, while numerous attributes are also discarded in the KIT profile.

2.3 CityJSON and the Extension mechanism

CityJSON is a JSON-based encoding for a subset of the CityGML data model, which in its current version (1.1.0) is conformant with CityGML version 3.0. The design decision behind CityJSON is to remove the deep hierarchical structure of the data model and to ensure an efficient way of storing geometries and semantics (Ledoux et al., 2019). Therefore, CityGML classes and CityObjects are mapped to CityJSON in a straightforward manner as long as they do not contradict with the main goal of CityJSON, which is to create a compact and user-friendly encoding. When this goal cannot be met, certain classes or concepts are either modified or completely removed.

Some of these differences are especially important for this MSc thesis since these concepts are utilized in Energy ADE as well, and may require special mapping rules to a CityJSON Extension. For instance, to remove the hierarchical structure, some abstract classes are not included in CityJSON. Additionally, boundary surfaces that are designed as CityObjects in CityGML are mapped to CityJSON as semantic objects with which primitives of a geometry may have a connection (Dukai and Ledoux, 2021).

Despite the fact that it is a rather new encoding, many studies can be found in the current literature that make use of CityJSON in various fields of study (Nys et al., 2020; Kippers et al.,

2021; Prataviera et al., 2021). Moreover, CityJSON provides an Extension mechanism similar to the ADE concept of CityGML which enables the addition of more specific information that is not already included in the core model. The main contrast between the two concepts is that while a CityGML ADE can be formulated in any way the user wants, CityJSON restricts the way the core model can be extended with three possible options (Ledoux et al., 2019):

1. New (complex) attributes can be added to existing CityObjects.
2. New CityObjects can be created (or existing ones can be extended), and complex geometries can be defined.
3. New properties can be added at the root of the document.

Considering that CityJSON lacks namespaces and inheritance, CityJSON Extensions are deemed to be less flexible than CityGML ADEs (Dukai and Ledoux, 2021). However, there are still numerous CityJSON extensions that successfully extend the core model. Nys et al. (2021) present a CityJSON extension that supports 3D point clouds by including MultiPoint as one of the allowed geometries of CityObjects. Vitalis et al. (2019) extend the data model of CityJSON by adding new root properties and additional attributes with the aim of providing a topological representation of 3D city models. Finally, Wu (2021) presents a CityJSON extension to store information about building permits by adding extra CityObjects and new attributes for existing *Building* and *LandUse* objects.

2.4 Space heating demand calculation and 3D city models

Two main approaches can be detected in Urban Energy Modelling for energy demand estimations: *top-down* and *bottom-up* approaches (Figure 2). The top-down approaches do not consider individual building parameters, but work with urban scale hypotheses about socio-economic, technical, or physical drivers (Prataviera et al., 2021). The bottom-up approaches take input data from individual or groups of houses to calculate energy demand, which are then aggregated for estimations on larger scales (Swan and Ugursal, 2009). While bottom-up approaches provide more detailed results, one drawback is that they require a large number of data that may not be publicly available in all cases (Ferrando and Causone, 2019).

Within bottom-up approaches, statistical and building simulation (physical) methods are highly used for energy demand estimations. Statistical methods use already measured energy consumption values to determine energy properties of building classes. These values are then used to estimate the energy demand of other buildings with similar properties (Kaden and Kolbe, 2014). Conversely, building simulation (physical) methods use simulation techniques, energy characteristics of buildings, and external data such as climate data to calculate energy demand. Due to the high number of input data requirements, building properties such as the typology and year of construction are used to categorize buildings, and the energy demand estimation is done for each of the categories (Ali et al., 2021).

Finally, building simulation methods can be classified as steady-state or dynamic models, depending on the level of detail of the used parameters. In steady-state models, fixed values are used for certain parameters such as solar radiation or air temperatures, and the climate data is taken in seasonal or monthly averages for simplification purposes (Dalla Mora et al., 2021). Therefore, the resulting energy demand calculations are usually presented as monthly or annual values. Contrarily, dynamic models present more detailed results since dynamic effects in buildings, as well as hourly climate data, are taken into account, and hourly or daily energy demand values can be obtained (Agugiaro et al., 2015). However, these models are usually preferred when detailed input parameters are available.

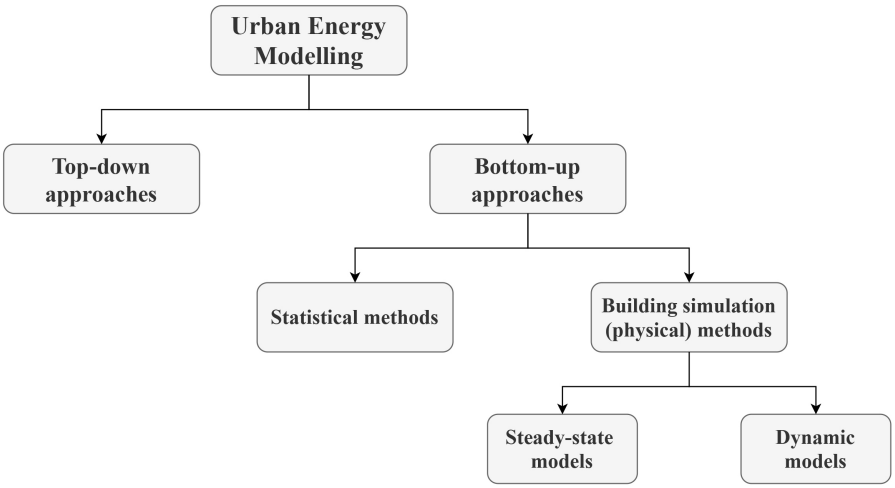

Figure 2: Various approaches in Urban Energy Modelling for energy demand estimations. Adapted from Swan and Ugursal (2009); Kaden and Kolbe (2014)

While numerous studies can be found about the aforementioned approaches and methods, the current literature suggests that steady-state models are favoured for city-scale space heating demand estimations, since less input parameters are needed and simplifications can be made for missing information. For this calculation, 3D city models are frequently used to obtain the input parameters about physical characteristics of buildings.

Agugiaro (2016b) uses an energy balance method that follows the Italian standards to calculate monthly space heating demand values for the city Trento, where many physical parameters such as the area, orientation, and pitch angles of building surfaces are obtained through the 3D city model of the city. Detailed parameters such as heat transfer coefficients and window-to-wall ratios are either obtained from existing building physics libraries or fixed values are assumed. Kaden and Kolbe (2014) use the 3D city model of Berlin in LoD2 to calculate similar input parameters for city-wide energy demand estimation of buildings according to the German standards based on an energy balance method.

Later studies also make use of the CityGML Energy ADE to calculate and store the input parameters, which are then used in heating demand simulations. Both Bruse et al. (2015) and Rossknecht and Airaksinen (2020) use Energy ADE on both input and output sides of the calculation. In these studies, the input parameters are first stored in the 3D city model with Energy ADE. Then, these parameters are used for heating demand simulations, and the resulting values are again stored by using Energy ADE.

3 Research questions

3.1 Objectives

This thesis aims to answer the following research question:

How can a CityJSON Energy Extension be used to support the calculation of space heating demand of buildings?

To answer this question, the following sub-questions are specified:

- To what extent is it possible to map CityGML Energy ADE classes to the CityJSON Energy Extension?

- Is a direct mapping from Energy ADE desirable? Are modifications needed?
- Which Energy ADE classes are related to space heating demand calculation?
- How can space heating demand calculation be used during the design phase to test the CityJSON Energy Extension?

3.2 Scope of research

This thesis will focus on the development of a CityJSON Energy Extension to be used for the calculation of space heating demand of buildings. The developed extension will ensure the storage of all input parameters as well as the resulting space heating demand values. Therefore, including classes and objects related to other energy applications is out of scope. Moreover, limitations of the CityJSON data model, its extensions, and its validator, *cjval*, will be investigated to provide recommendations to these projects. In addition, space heating demand will be calculated with a steady-state model based on an energy balance method while dynamic models are out of the scope of this research.

4 Methodology

The proposed methodology consists of two main steps: development of the CityJSON Energy Extension and the calculation of space heating demand of buildings as the use case (Figure 3). Even though the two steps are described separately in the following subsections, it is important to keep in mind that they are highly interrelated since the use case is utilized continuously during the development of the CityJSON Energy Extension to test its functionalities.

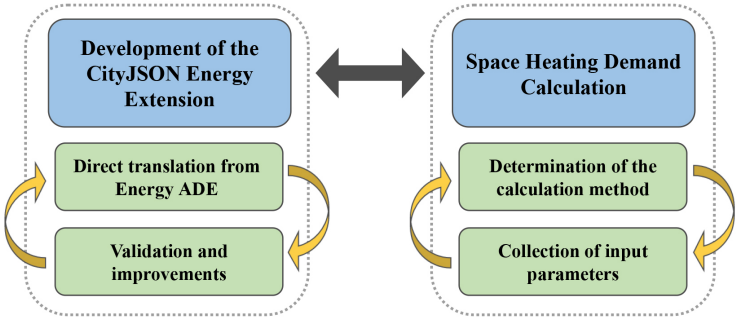

Figure 3: Overview of the proposed methodology.

4.1 Development of the CityJSON Energy Extension

4.1.1 Direct translation from Energy ADE to a CityJSON Energy Extension

The first step includes creating a direct translation from Energy ADE to a CityJSON Energy Extension to grasp the similarities and differences between CityGML and CityJSON data models. Because of its lower complexity compared to the full Energy ADE, the KIT profile is chosen for this purpose. The following aspects will be taken into account when designing the direct translation:

- Mapping of new CityObjects as well as new non-CityObjects as described in the Energy ADE.
- Addition of new data types, enumerations and code lists.

- Inheritance relationships and associations among CityObjects and non-CityObjects.
- Extension of existing classes with new attributes and properties.

While a straightforward mapping is aimed in this step, additional mapping rules will be considered if a direct translation is not possible due to the restrictions of CityJSON's extension mechanism. In this case, JSON schema will be investigated to identify useful objects and functionalities that may help translate certain Energy ADE concepts.

4.1.2 Validation and improvements

Once a direct translation from the Energy ADE KIT Profile to a CityJSON Energy Extension is obtained, the validation of the extension will be done in two steps. Firstly, the use case, the calculation of space heating demand of buildings, will be utilized to create a CityJSON + Energy Extension file with the readily available (or obtainable) input data needed for this use case. It will be evaluated whether it is possible to store this information in a straightforward manner in the developed Energy Extension or not. Secondly, the official validator of CityJSON, *cjval* [1], will be used to inspect the validity of the CityJSON + Energy Extension file, which provides a validation against official CityJSON schemas as well as additional functionalities that are described in Table 1. This way, the Energy Extension schema will also be validated.

It is important to note that the validity of the CityJSON Energy Extension against CityGML Energy ADE is not considered in this research. The reason for this is that even though the first step is to create a direct mapping from Energy ADE to CityJSON Energy Extension, it is not the approach that will be used in the next steps considering the diverse philosophies behind CityGML and CityJSON data models, as well as the capabilities of XML and JSON data formats. Since the concepts and functionalities of the CityJSON Energy Extension may differ from Energy ADE in the final stage, it was decided not to use this validation method but to focus on the use case to validate the extension.

***Cjval* function**	**What it does**
JSON syntax	Checks the file against the JSON schema
CityJSON schemas	Checks the file against CityJSON schemas (v1.1)
Extension schemas	If an Extension is present in the input file, it is validated against the Extension schema of CityJSON
Parent/children consistency	Checks whether the referenced parents/children actually exist
Wrong vertex index	Vertices list and their indices are checked
Semantics array	Checks if semantics of a geometry are added correctly
Extra root properties	Gives a warning if an extra root property is not defined in an Extension
Duplicate vertices	Gives a warning if there are duplicate vertices
Unused vertices	Gives a warning if there are unreferenced vertices

Table 1: Validation functionalities of *cjval* [1]

[1] https://github.com/cityjson/cjval

After obtaining a valid direct mapping, the next step focuses on making improvements on the Energy Extension based on the use case. To provide a complete space heating demand calculation, the urban area within the Rijssen-Holten municipality in the Netherlands is chosen as the study area. All input data needed for this calculation as well as the resulting energy demand values will be stored in the CityJSON Energy Extension. To improve the extension, the storage efficiency and the ease of retrieving data from the CityJSON + Energy Extension file will be assessed. Moreover, if it is detected that the CityJSON Energy Extension lacks certain concepts or classes related to the use case, these will be included as well.

4.2 Space heating demand calculation

4.2.1 Calculation method

Space heating demand calculation will be performed based on the determination method specified by the Dutch standard, *NTA 8800*, since the chosen study area is located in the Netherlands. The aim of this standard is to provide a determination method for calculating the energy performance of buildings with a focus on the energy requirement, primary fossil energy use, renewable energy, and the net heat demand of a building (NEN, 2020).

The calculation is based on a steady-state energy balance method, which takes monthly average values of input parameters, and the dynamic effects are considered with utilization factors (Radwan, 2021). According to *NTA 8800*, the net monthly energy demand of a building for space heating is calculated as follows;

$$Q_{H;nd;zi;mi} = (Q_{H;tr;zi;mi} + Q_{H;ve;zi;mi}) - n_{H;gn;zi;mi}(Q_{H;int;zi;mi} + Q_{H;sol;zi;mi})$$

where the left side of the formula shows the heat losses through transmission ($Q_{H;tr;zi;mi}$) and ventilation ($Q_{H;ve;zi;mi}$) and the right side represents the heat gains through internal ($Q_{H;int;zi;mi}$) and solar ($Q_{H;sol;zi;mi}$) gains multiplied by a dimensionless utilization factor ($n_{H;gn;zi;mi}$) to consider dynamic effects. Explanation of each element as well as some of the needed input parameters are illustrated in Table 2. Additionally, since this calculation will be done on city-scale and not for single buildings, it will not be possible to obtain all needed parameters such as the frame fraction of windows and number of vertical pipes. In these cases, simplifications and assumptions will be made so that the calculation can be performed for the whole area.

4.2.2 Collection of input parameters

As described before, space heating demand calculation requires a high number of parameters; therefore, different sources will be investigated to collect all needed input data. Firstly, the 3D city model will be used to perform geometrical calculations to obtain the volume of the building, area of the thermal boundaries, and to infer the number of storeys, which can then be used to classify buildings into typical building classes (Agugiaro, 2016b). Then, building physics libraries will be used to obtain more detailed parameters, such as the thermal transmittance of opaque surfaces, window-to-wall ratios, and solar gain factors of windows. The European project TABULA [2] will be used as the main source for this, which classifies residential buildings in each European country into building typologies that contain specific building physics properties. Finally, any other publicly accessible dataset from Rijssen-Holten will be obtained, and previous related studies (Agugiaro, 2016b; Rossknecht and Airaksinen, 2020; Kaden and Kolbe, 2013; Bruse et al., 2015) will be used when simplifications and assumptions need to be made.

[2] https://episcope.eu/iee-project/tabula/

Element	Explanation	Main input parameters
Heat losses through transmission ($Q_{H;tr;zi;mi}$)	Considers the heat transfer (1) between the heated space and outside air, (2) via adjacent unheated spaces, (3) via adjacent heated spaces, (4) through vertical pipes passing through the thermal envelope and in contact with outside air	- Thermal transmittance of opaque surfaces - Surface areas - Number of storeys - Number of vertical pipes - Temperature of the calculation zone and outside air
Heat losses through ventilation ($Q_{H;ve;zi;mi}$)	Considers the ventilation air that enters the calculation zone and causes heat losses	- Heat capacity of air - Volume of effective air flow - Temperature of the calculation zone and outside air
Internal heat gains ($Q_{H;int;zi;mi}$)	Considers the internal heat produced by people and the facilities located in the calculation zone	- Number of residential units in the calculation zone - The average number of residents per residential unit
Solar gains ($Q_{H;sol;zi;mi}$)	Considers the on-site solar radiation, orientation, sun absorption and heat transfer properties of the receiving transparent and opaque surfaces	- Area, frame fraction and solar gain factor of windows - Incident solar radiation - Heat transfer resistance of opaque surfaces

Table 2: Elements of the space heating demand calculation and needed input parameters. Adapted from NEN (2020)

5 Preliminary results

5.1 Direct translation of the KIT Profile

To assess the similarities and differences between CityGML and CityJSON, a CityJSON Energy Extension was created as a direct mapping from the Energy ADE KIT Profile without focusing on the efficiency of the Extension. Moreover, when a direct mapping is not possible, preliminary design decisions were made to be able to obtain a complete Energy Extension for testing purposes. The main mapping rules are explained in the following subsections and examples from the JSON schema as well as the corresponding exemplary JSON objects are given from a subset of the Core module of the KIT Profile (Figure 4). However, not only the Core module elements but the whole KIT profile was translated with the specified mapping rules.

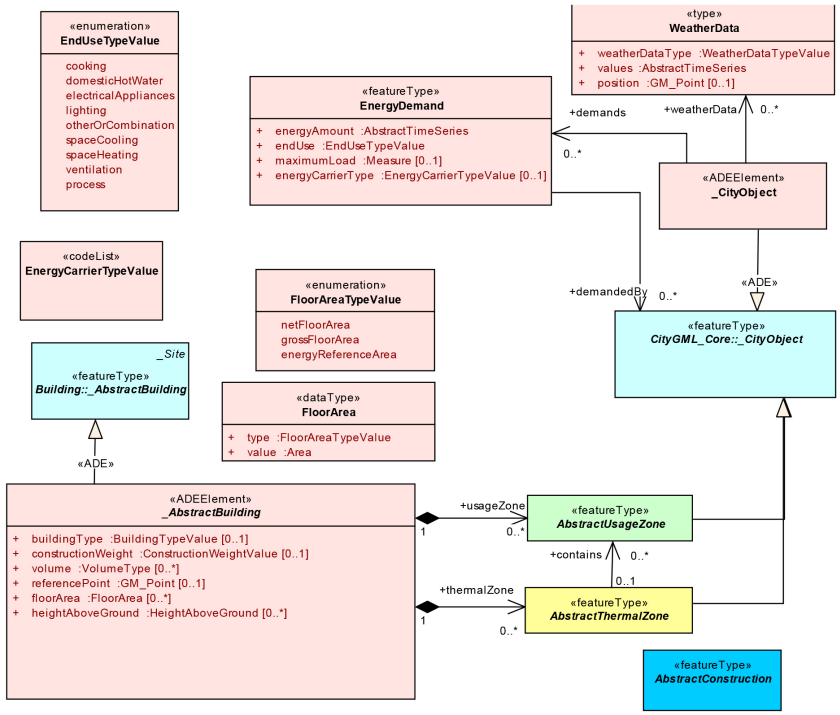

Figure 4: A subset of the Core module of Energy ADE KIT Profile (Benner, 2018).

5.1.1 Creating new CityObjects

CityJSON's Extension mechanism allows the addition of new CityObjects with their attributes by using the "extraCityObjects" member, which was used to map the new CityObjects in the KIT Profile. However, a design decision was made about inheritance, since this is not supported in CityJSON. According to the CityGML data model, all CityObjects are subclasses of the *AbstractCityObject* class (Gröger et al., 2012). To ensure a similar mechanism, the "allOf" keyword of JSON schema was used which ensures that a schema is also valid against all of the given subschemas [3]. An example is given below, where a new "UsageZone" object is defined in the schema that references *AbstractCityObject* (left), and the corresponding examplary JSON object is shown (right).

```
"extraCityObjects": {
  "+UsageZone": {
    "allOf": [
      {
      "$ref":"cityobjects.
         schema.json#/
         _AbstractCityObject"
      },
      {
        "properties": {...}
      }
    ]
  }
}
```

```
"Usage1": {
  "type": "+UsageZone",
  "attributes": {
    "usageZoneType": "
       residential",
    "floorArea": {
      "type": "netFloorArea",
      "value": {
        "value": 100,
        "uom": "m2"
      }
    }
  }
}
```

[3] https://json-schema.org/

It was also decided not to map new abstract CityObjects defined in the KIT Profile to the CityJSON Energy Extension. The reason for this is that in the current version of Energy ADE, the abstract base classes (e.g., *AbstractUsageZone*, *AbstractThermalZone*) do not include any properties to pass on to their subclasses. Following the philosophy of CityJSON, these abstract classes were ignored to avoid deep hierarchical structures.

5.1.2 New attributes to existing CityObjects

CityJSON's Extension mechanism allows the addition of new attributes to existing CityObjects with the "extraAttributes" member. An example is given below where the existing *Building* CityObject was extended with new attributes as defined in the KIT Profile.

```
"extraAttributes": {
  "Building": {
    "+buildingType": {...},
    "+constructionWeight":
       {...},
    "+volume": {...},
    "+floorArea": {...},
    "+heightAboveGround": {...}
  }
}
```

```
"Build1": {
  "type": "Building",
  "geometry": [...],
  "attributes": {
    "+buildingType": "
       singleFamily",
    "+constructionWeight": "
       heavy",
  }
}
```

5.1.3 Creating new non-CityObjects

The Energy ADE KIT profile contains not only new CityObjects, but also new classes that are not derived from *AbstractCityObject*. However, a direct mapping of these classes to the CityJSON Energy Extension is not possible since the Extension mechanism only supports the addition of new CityObjects. Therefore, it was decided to create these classes under "extraCityObjects" without using the "allOf" keyword to reference *AbstractCityObject*. For instance, the EnergyDemand class defined in the Core module of the KIT profile is not a CityObject but is derived from *gml:AbstractFeatureType*; therefore, it was mapped as follows:

```
"extraCityObjects": {
  "+EnergyDemand": {
    "type": "object",
    "properties": {
      "type": {...},
      "attributes": {
        "type": "object",
        "properties": {
          "energyAmount": {...},
          "endUse": {...},
          "maximumLoad": {...},
          "energyCarrierType":
             {...}
        }
    ...
}
```

```
"Demand1": {
  "type": "+EnergyDemand",
  "attributes": {
    "endUse": "spaceHeating",
    "energyCarrierType": "
       naturalGas",
    "energyAmount": [...], //ID
        of TimeSeries object
    "maximumLoad": {
      "value": 250,
      "uom": "kWh/m2"
    }
  }
}
```

Similarly, all objects with the ≪ *type* ≫ stereotype are derived from *gml:AbstractGMLType*, such as the classes of Time Series, Schedule and Weather Data in the KIT Profile, and were therefore mapped as "extraCityObjects" as well. It can be argued that even though this approach is used for the preliminary tests, it is not the most ideal solution since it is confusing to include both CityObjects and non-CityObjects within the "extraCityObjects" member. Therefore, the improvements of the direct mapping will consider this problem in the next steps.

5.1.4 New data types, enumerations and code lists

The KIT Profile defines specific data types, enumerations, or code lists to be used as the allowed values of certain attributes. In the CityJSON Energy Extension, data types are defined as subschemas under "definitions" so that they can be referenced multiple times. Moreover, the "enum" keyword of the JSON schema is used to restrict the value of an attribute to a fixed set of values [4]. An example is given below, where the *floorArea* data type is defined under "definitions", which itself contains an enumeration.

```
"definitions": {
  "floorArea": {
    "type": "object",
    "properties": {
      "type": {
        "enum": ["netFloorArea",
            "grossFloorArea", "
           energyReferenceArea"]
      },
      "value": {...}
    }
  }
}
```

```
"Build1": {
  "type": "Building",
  "geometry": [...],
  "attributes": {
    "+buildingType": "
       singleFamily",
    ...
    "+floorArea": {
      "type": "netFloorArea",
      "value": {...}
    }
  }
}
```

Unlike enumerations, code lists were not mapped to the CityJSON Energy Extension since these values may not be fixed in advance but may be determined by the responsible authorities. Therefore, when the data type is a code list in the KIT Profile, these were mapped as "string" types.

5.1.5 Relations among CityObjects and non-CityObjects

The relations among CityObjects and non-CityObjects were handled as two separate cases. If there is a relation between two CityObjects, the "parents/children" concept of JSON was used to make the connection between them. For instance, Figure 4 shows that a ThermalZone may contain 0 or more UsageZones. To ensure this relationship, parents and children properties were defined to specify the ID(s) of the corresponding parent or children. An example is shown below for the ThermalZone object, and the corresponding ThermalZone and UsageZone objects are illustrated on the right where the "parents/children" properties refer to the IDs of the two objects.

[4] https://json-schema.org/understanding-json-schema/reference/generic.html

```
"+ThermalZone": {
  "type": {...},
  "properties": {
    "parents": {
      "type": "array",
      "items": {"type":"string"}
    },
    "children": {
      "type": "array",
      "items": {"type":"string"}
    }
  }
}
```

```
"Zone1": {
  "type": "+ThermalZone",
  ...,
  "children": ["Usage1"]
},
"Usage1": {
  "type": "+UsageZone",
  "attributes": {
  "usageZoneType":"residential"
  },
  "parents": ["Zone1"],
  ...
}
```

If, however, there is a relation between a CityObject and non-CityObject, or between two non-CityObjects, an additional attribute was added to the corresponding object to store the IDs of the objects that it has a relation with. For instance, Figure 4 demonstrates that all CityObjects have a "+demands" relation with the EnergyDemand object. To ensure this relation, an additional attribute called "energyDemand" was added to all CityObjects, and an example is given below for the case of the Building CityObject.

```
"extraAttributes": {
  "Building": {
    "+buildingType": {...},
    ...,
    "+energyDemand": {
      "type": "array",
      "items": {"type":"string"}
    //ID of EnergyDemand objects
    }
  }
}
```

```
"Build1": {
  "type": "Building",
  "geometry": [...],
  "attributes": {
    "+buildingType": "
       singleFamily",
    ...
    "+energyDemand":["Demand1"]
  }
}
```

The reason of using a different mechanism for non-CityObjects is that these relations are simply connections between various objects and not parents/children relationships. Therefore, these relations were added with additional attributes whenever required in the KIT Profile.

5.2 Tests with existing input data for space heating demand calculation

To test the direct translation, existing energy-related data from Rijssen-Holten was stored with the CityJSON Energy Extension. The input data is originally stored as *Generic Attributes* in CityGML format, where for each surface the azimuth, direction, inclination, area in LoD2, and the surface normal are present (Figure 5).

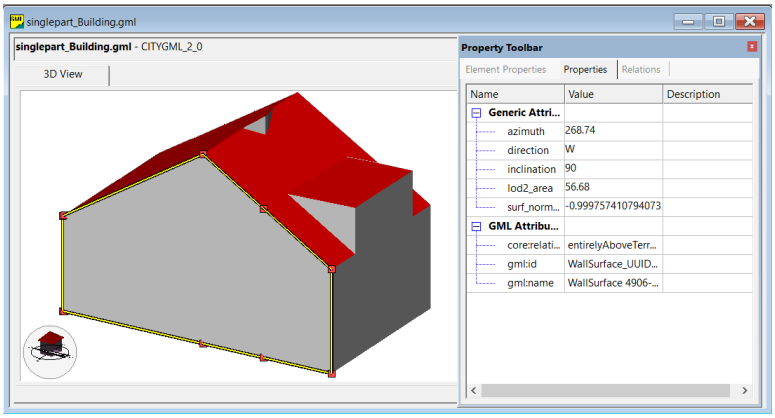

Figure 5: Existing input data from Rijssen-Holten: energy-related surface properties of buildings.

Firstly, the CityGML file was converted to CityJSON with *citygml-tools* [5], a command line utility for processing CityGML files. Since *citygml-tools* currently supports CityJSON version 1.0.3, the file was then upgraded to version 1.1.0 with *cjio* [6], which is a Python command-line interface to process and manipulate CityJSON files. Finally, two steps were taken to store the input data in a CityJSON + Energy Extension file. For each building, a ThermalZone object was created. Then, for each ThermalZone object, ThermalBoundary objects were created that consisted of Ground, Wall and Roof surfaces. All these objects were created without a geometry since the ThermalBoundary and ThermalZone objects refer to their corresponding Building or BuildingPart with the parents/children relationship. Finally, surface type, azimuth, inclination, and area information were stored as attributes of each ThermalBoundary while direction and surface normal attributes were omitted since these attributes are not related to the space heating demand calculation and are not supported in the KIT Profile (Figure 6). The resulting file was then validated with *cjval*.

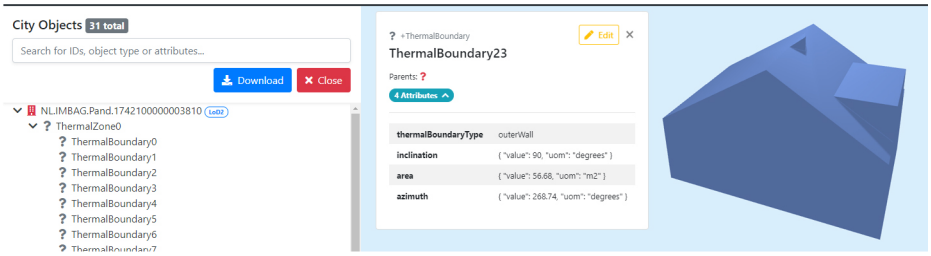

Figure 6: Energy-related data stored in CityJSON + Energy Extension file.

6 Time planning

Time planning for the thesis and the corresponding tasks are shown in Figure 7. Writing will be done together with the other tasks to be able to develop the project in a parallel way.

[5] https://github.com/citygml4j/citygml-tools
[6] https://github.com/cityjson/cjio

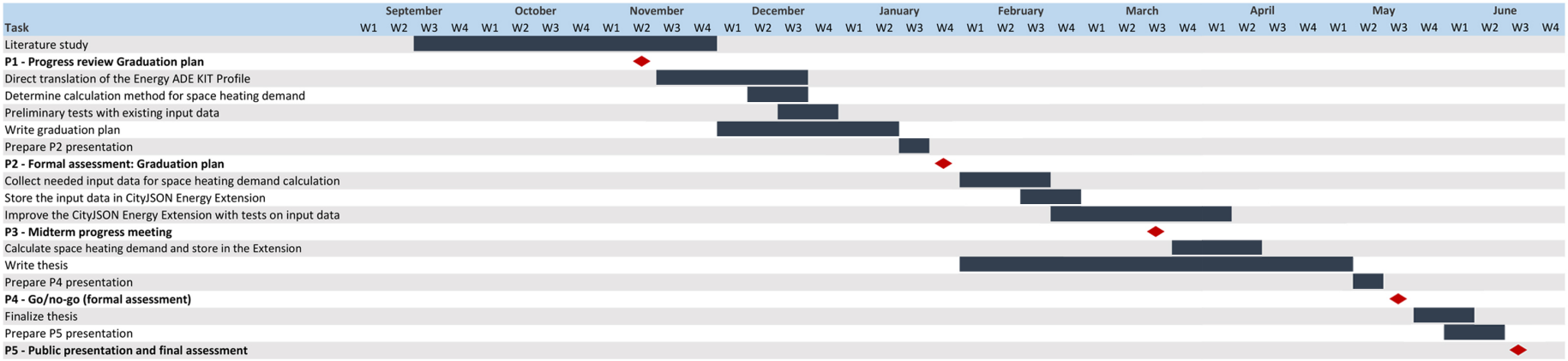

Figure 7: Time planning for the thesis

7 Tools and datasets used

7.1 Tools

To create the CityJSON Energy Extension and exemplary CityJSON files for testing, PyCharm is used as the development environment. CityJSON files are visualised in *ninja* [7], and their validity is checked with *cjval* [8], the official validator of CityJSON. Moreover, *cjio* [9], a Python command-line interface to process and manipulate CityJSON files, is used to filter LoDs or to upgrade the CityJSON version. For the calculation of space heating demand, existing input data in CityGML format is converted to CityJSON with *citygml-tools* [10], which is a command line utility for processing CityGML files. Finally, Python is selected as the programming language to be used for the processing of input data, calculation of parameters from the 3D city model, and to store these input parameters, as well as the resulting space heating demand values in the CityJSON + Energy Extension file.

7.2 Datasets

For the calculation of space heating demand of buildings, existing 3D city model of Rijssen-Holten in CityGML format will be used to obtain certain input parameters. This dataset was created as a testbed for energy applications (León-Sánchez et al., 2021), which is in its beta version at the moment and will be publicly available in the near future. It includes the building footprints in LoD0 as well as the building geometries in LoD2, and attributes about the number of storeys, year of construction, volume, and building classes are included. Additionally, each surface is enriched with additional information such as the inclination, surface area, surface normal, azimuth, and direction. In addition to the existing input parameters that can be collected from this dataset, the 3D city model will also be used to perform geometrical calculations to obtain thermal zones and shared walls.

The Dutch building physics library from the TABULA project will be used to obtain building physics parameters such as the heat capacity, conductivity, U-values, and g-values of the buildings' surfaces. This data is provided in the TABULA WebTool [11] where the values are differentiated by the building type (single family, terraced, apartment block, etc.) and the year of construction.

Current energy labels of the buildings in Rijssen-Holten will be used to verify the results of the space heating demand calculation. This data is publicly available on EP-Online, the official national database to register energy performance indicators and energy labels [12]. It is possible through this database to obtain monthly energy label information for a specific area such as on address, district, municipality, or province level.

[7] https://ninja.cityjson.org/
[8] https://github.com/cityjson/cjval
[9] https://github.com/cityjson/cjio
[10] https://github.com/citygml4j/citygml-tools
[11] https://webtool.building-typology.eu
[12] https://www.ep-online.nl/

References

G. Agugiaro. Enabling "energy-awareness" in the semantic 3D city model of Vienna. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-4/W1, Sept. 2016a. doi: 10.5194/isprs-annals-IV-4-W1-81-2016.

G. Agugiaro. Energy planning tools and CityGML-based 3D virtual city models: experiences from Trento (Italy). *Applied Geomatics*, 8(1):41–56, 2016b.

G. Agugiaro and P. Holcik. *3D City Database extension for the CityGML Energy ADE 0.8 (PostgreSQL Version)*, Sept. 2017.

G. Agugiaro, S. Hauer, and F. Nadler. Coupling of CityGML-based semantic city models with energy simulation tools: some experiences. pages 191–200. REAL CORP 2015 Proceedings/-Tagungsband, May 2015. ISBN 978-3-9503110-8-2.

G. Agugiaro, J. Benner, P. Cipriano, and R. Nouvel. The Energy Application Domain Extension for CityGML: enhancing interoperability for urban energy simulations. *Open Geospatial Data, Software and Standards*, 3(1):1–30, 2018.

U. Ali, M. H. Shamsi, C. Hoare, E. Mangina, and J. O'Donnell. Review of urban building energy modeling (UBEM) approaches, methods and tools using qualitative and quantitative analysis. *Energy and Buildings*, 246:111073, Sept. 2021. ISSN 0378-7788. doi: 10.1016/j.enbuild.2021.111073. URL https://www.sciencedirect.com/science/article/pii/S0378778821003571.

J. Benner. *CityGML Energy ADE V. 1.0 Specification*, 2018. URL http://www.citygmlwiki.org/images/3/38/Energy_ADE_specification_2018_03_25.pdf.

P. v. d. Brom. *Energy in Dwellings: A comparison between Theory and Practice*. PhD thesis, Feb. 2020. URL https://journals.open.tudelft.nl/abe/article/view/4664.

M. Bruse, R. Nouvel, P. Wate, V. Kraut, and V. Coors. An Energy-Related CityGML ADE and Its Application for Heating Demand Calculation. *International Journal of 3-D Information Modeling*, 4:59–77, July 2015. doi: 10.4018/IJ3DIM.2015070104.

T. Dalla Mora, L. Teso, L. Carnieletto, A. Zarrella, and P. Romagnoni. Comparative Analysis between Dynamic and Quasi-Steady-State Methods at an Urban Scale on a Social-Housing District in Venice. *Energies*, 14(16):5164, Jan. 2021. doi: 10.3390/en14165164. URL https://www.mdpi.com/1996-1073/14/16/5164.

B. Dukai and H. Ledoux. CityJSON Specifications 1.1.0, Dec. 2021. URL https://www.cityjson.org/specs/1.1.0/.

M. Ferrando and F. Causone. An overview of urban building energy modelling (UBEM) tools. In *16th IBPSA International Conference and Exhibition*. International Building Performance Simulation Association, Sept. 2019. doi: 10.26868/25222708.2019.210632.

G. Gröger, T. H. Kolbe, C. Nagel, and K.-H. Häfele. *OGC City Geography Markup Language (CityGML) Encoding Standard*, 2012. URL https://portal.ogc.org/files/?artifact_id=47842.

G. Gröger and L. Plümer. CityGML – Interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71:12–33, July 2012. ISSN 0924-2716. doi: 10.1016/j.isprsjprs.2012.04.004. URL https://www.sciencedirect.com/science/article/pii/S0924271612000779.

R. Kaden and T. Kolbe. City-wide total energy demand estimation of buildings using semantic 3D city models and statistical data. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-2/W1:163–171, Sept. 2013. doi: 10.5194/isprsannals-II-2-W1-163-2013.

R. Kaden and T. H. Kolbe. Simulation-based total energy demand estimation of buildings using semantic 3D city models. *International Journal of 3-D Information Modeling (IJ3DIM)*, 3 (2):35–53, 2014.

R. Kippers, M. Koeva, M. Keulen, and S. Oude Elberink. Automatic 3D Building Model Generation Using Deep Learning Methods Based On CityJSON and 2D Floor Plans. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVI-4/W4-2021:49–54, Oct. 2021. doi: 10.5194/isprs-archives-XLVI-4-W4-2021-49-2021.

H. Ledoux, K. A. Ohori, K. Kumar, B. Dukai, A. Labetski, and S. Vitalis. CityJSON: A compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4(1):1–12, 2019.

C. León-Sánchez, D. Giannelli, G. Agugiaro, and J. Stoter. Testing the New 3D BAG Dataset for Energy Demand Estimation of Residential Buildings. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVI-4/W1-2021:69–76, Sept. 2021. doi: 10.5194/isprs-archives-XLVI-4-W1-2021-69-2021.

D. Majcen. *Predicting energy consumption and savings in the housing stock: A performance gap analysis in the Netherlands*. PhD thesis, 2016. URL `https://repository.tudelft.nl/islandora/object/uuid%3A00795500-6704-49c0-903f-6b6ba77dc544`.

NTA 8800:2020+A1:2020 nl. NEN, 2020. URL `https://www.nen.nl/en/nta-8800-2020-a1-2020-nl-278296`.

G.-A. Nys, F. Poux, and R. Billen. CityJSON Building Generation from Airborne LiDAR 3D Point Clouds. *ISPRS International Journal of Geo-Information*, 9(9):521, Sept. 2020. doi: 10.3390/ijgi9090521. URL `https://www.mdpi.com/2220-9964/9/9/521`.

G.-A. Nys, A. Kharroubi, F. Poux, and R. Billen. An Extension of CityJSON to Support Point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42:301–306, 2021.

E. Prataviera, P. Romano, L. Carnieletto, F. Pirotti, J. Vivian, and A. Zarrella. EUReCA: An open-source urban building energy modelling tool for the efficient evaluation of cities energy demand. *Renewable Energy*, 173:544–560, Aug. 2021. ISSN 0960-1481. doi: 10.1016/j.renene.2021.03.144. URL `https://www.sciencedirect.com/science/article/pii/S0960148121005085`.

Y. M. G. Radwan. Automation of Building Energy Efficiency Using BIM: A decision support BIM-based tool to optimize the EI of buildings. Master's thesis, 2021. URL `https://research.tue.nl/en/studentTheses/automation-of-building-energy-efficiency-using-bim`.

M. Rossknecht and E. Airaksinen. Concept and Evaluation of Heating Demand Prediction Based on 3D City Models and the CityGML Energy ADE—Case Study Helsinki. *ISPRS International Journal of Geo-Information*, 9(10):602, Oct. 2020. doi: 10.3390/ijgi9100602. URL `https://www.mdpi.com/2220-9964/9/10/602`.

L. G. Swan and V. I. Ugursal. Modeling of end-use energy consumption in the residential sector: A review of modeling techniques. *Renewable and Sustainable Energy Reviews*, 13(8): 1819–1835, Oct. 2009. ISSN 1364-0321. doi: 10.1016/j.rser.2008.09.033. URL https://www.sciencedirect.com/science/article/pii/S1364032108001949.

United Nations Environment Programme. 2020 Global Status Report for Buildings and Construction: Towards a Zero-Emission, Efficient and Resilient Buildings and Construction Sector. Technical report, 2020.

S. Vitalis, K. Arroyo Ohori, and J. Stoter. Incorporating Topological Representation in 3D City Models. *ISPRS International Journal of Geo-Information*, 8(8), 2019. ISSN 2220-9964. doi: 10.3390/ijgi8080347. URL https://www.mdpi.com/2220-9964/8/8/347.

J. Wu. Automatic building permits checks by means of 3D city models. Master's thesis, 2021. URL https://repository.tudelft.nl/islandora/object/uuid%3Ad6d23631-281a-4107-9f05-7e4942f2dabf.

Z. Yao, C. Nagel, F. Kunde, G. Hudra, P. Willkomm, A. Donaubauer, 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. ISSN 2363-7501. doi: 10.1186/s40965-018-0046-7. URL https://doi.org/10.1186/s40965-018-0046-7.