OGC Indoor Mapping and Navigation Pilot

Public Safety Features CityGML ADE ER

Publication Date: 2020-07-30

Approval Date: 2019-11-21

Submission Date: 2019-06-03

Reference number of this document: OGC 19-032

Reference URL for this document: http://www.opengis.net/doc/PER/NIST-PSF-CityGMLade

Category: OGC Public Engineering Report

Editor: Steven Chau & Mohsen Kalantari

Title: OGC Indoor Mapping and Navigation Pilot: Public Safety Features CityGML ADE ER

OGC Public Engineering Report

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the

Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Table of Contents

Chapter 1. Summary

This document defines an Application Domain Extension (ADE) of CityGML for public safety use cases. The ADE has been developed as part of OGC's Indoor Mapping and Modeling Pilot project [http://www.opengeospatial.org/projects/initiatives/indoor-pilot] sponsored by the National Institute of Standards and Technology (NIST), Communications Technology Laboratory (CTL), Public Safety Communications Research (PSCR) Division. The ADE has been developed primarily based on reference preplan symbology created by the National Alliance for Public Safety GIS (NAPSG) Foundation [https://www.napsgfoundation.org/]. NAPSG is a 501 (C) (3) not-for-profit organization that was established in 2005 to overcome challenges faced by Federal, tribal, state, and local public safety agencies in the United States. NAPSG focuses on using GIS technology to resolve challenges that occur. In the definition of the ADE, public safety requirements that were not explicit in NAPSG have also been considered. This Engineering Report (ER) provides the methodology of the ADE development, details the implementation of the ADE and its structure and the application of the ADE in the context of public safety use cases.

The findings include:

- A methodology to transform NAPSG symbology to data elements;
- A need for an extension of a reference to four existing CityGML classes; and
- The creation of seven new CityGML classes that are critical for public safety use cases.

1.1. Requirements & Research Motivation

Preparation is a critical aspect of public safety through which the state of the locations, places, venues, buildings and infrastructures, and associated facilities and services are documented, so responses to an emergency are informed and facilitated. Response to a crisis in indoor environments requires information beyond the architectural layout of interiors. For example, in preparation for safety, architectural components such as openings need to be more specifically defined concerning their safety functions; e.g., as an emergency exit or smoke door. It is also required that the location and specification of safety features such as sensors and detectors in the indoor environment to be defined. This report aims to build on the foundation CityGML provides for modeling architectural components of interiors and extend it for the requirements of public safety.

Additionally, NAPSG Foundation's goal of equipping GIS responders and public safety practitioners with the knowledge and skillset needed aligned with the objective of this ADE. This effort used NAPSG's symbology due to the expertise of the NAPSG stakeholders.

1.2. Prior-After Comparison

The public safety ADE features an extension of CityGML concerning safety in the indoor environment. In this ADE, existing features such as Room, IntBuildingInstallation, Door, Window have been extended through which new public safety feature types and associated attributes are created. A new feature type called PublicSafetyHatch that has been created based on an existing abstract CityGML feature type AbstractOpening.

1.3. Recommendations for Future Work

This ER has been developed at the early stages of the Indoor Mapping and Modeling pilot project based on the methodology that only relies on document analysis, expert opinion and generating instance CityGML files. While it is confirmed that the ADE is structurally valid and functional to create CityGML compliant files, it is recommended that the ADE developed in this ER to be tested in public safety use case scenarios so the utility of it can be examined in practice. Also, in preparing this ADE, it is realized that there are limited official guides to create CityGML ADEs. This gap and the scattered knowledge base will impede the adoption of the CityGML ADE in domains where geospatial data is not core, but its need is being increasingly realized.

1.4. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Contacts

Name	Organization
Mohsen Kalantari	Faramoon
Steven Chau	Faramoon
Don Withange	Faramoon

1.5. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

Chapter 2. References

The following normative documents are referenced in this document.

- OGC 12-019, OGC City Geography Markup Language (CityGML) Encoding Standard [https://portal.opengeospatial.org/files/?artifact_id=47842]

Chapter 3. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

- Symbology

Symbology refers to visual symbols the NAPSG uses to illustrate and categorize hazards.[1]

3.1. Abbreviated terms

- ADE Application Domain Extensions
- CityGML City Geography Markup Language
- GIS Geographic Information System
- GML Geography Markup Language
- NAPSG National Alliance for Public Safety GIS Foundation
- UML Unified Modeling Language
- URI Uniform Resource Identifier
- XML Extensible Markup Language
- XSD XML Schema Definition

[1] For examples see: https://napsg-web.s3.amazonaws.com/symbology/index.html#/

Chapter 4. Overview

This ER is provided in seven sections.

Section 1 provides a summary of the ER including motivation, methodology, outputs, and outcomes regarding the Public Safety ADE.

Section 2 lists the reference documents used in the development of the ADE and preparation of the ER.

Section 3 lists the abbreviated terms used throughout the ER.

Section 4, this section, outlines the structure of this ER.

Section 5 documents the process of developing the PublicSafety ADE. It describes a step-by-step process on how it was developed, as well as challenges and difficulties.

Section 6 analyzes the PublicSafety ADE XSD and how it has been represented in the UML diagram. The section thoroughly explains how the XSD code works and explains the UML diagram structures.

Section 7 provides a conclusion of the PublicSafety ADE development and also suggestions for future directions.

Chapter 5. Methodology

NOTE

Summary

This section explains the methodology to develop the Public Safety ADE.

The methodology was separated into several steps. The first step involved an analysis of the NAPSG symbology. The second step comprised the identification of public safety features and their properties. In the third step, the CityGML ADE was analyzed to select the most appropriate approach for creating the Public Safety ADE. The third step was followed by identifying the types of relationships the public safety features required to make with CityGML features. The ADE was modeled in UML and then a schema file was created from the UML diagram. The ADE was evaluated by validating the schema and utilizing it as a sample file.

5.1. Step 1. Analysis of the NAPSG symbology

NOTE

Summary

This section explains how the public safety symbology was analyzed for the development of the ADE.

5.1.1. Acquiring the public safety symbology

To identify the public safety features, symbology from the NAPSG Foundation was analyzed with a particular focus on symbology in the Preplan section, which includes 217 symbols. The symbols have been categorized into groups as follows:

- Preplan Features
- Fire Suppression
- Alarm and Detectors
- Feature Shutoff
- Vertical Access Features
- Key or Knox Box

These symbols then were individually analyzed based on the following criteria.

1. If the symbol is related to indoors, outdoors, or both.
    a. If the symbol is related to indoors or both, the symbology will be put into consideration to be used in the ADE.
    b. If the symbol is related to outdoors, the symbology will not be put into consideration to be used in the ADE.
2. If the symbol is clearly defined and/or obvious.
    a. For example, a symbol representing a sprinkler is clearly defined and obvious.
    b. For example, some unclear and ambiguous symbols were “Fire Department Connection, FDC1” and “Fire Department, FDC2.”

i. Expert assistance was required for this group of symbols.

3. If the symbol is a feature, attribute, both, or unknown.
   a. If the symbol is a feature, it will likely have its own class in the ADE.
   b. If the symbol is an attribute, it will likely be an attribute of a class in the ADE.
   c. If the symbol could be both, it will be reviewed based on numerous factors.
      i. Most symbols were considered both feature and attribute. This was due to scenarios like the following example:
         A. There is a symbol for “Heat Detector.” There are also numerous symbols for Heat Detector subtypes such as Flame, Gas, and Smoke. Heat Detector could be a feature because it has many subtypes. However, there is a Detector symbol as well. This symbol also categorizes other symbols such as Gas Detector and Smoke Detector. Therefore, Heat Detector was determined to be an attribute of Detector.
      ii. Another factor required looking at the existing CityGML UML diagram to deduce the best method of implementing the symbol.
      iii. The other factor was more clarification was required to know the most appropriate method to implement the symbol in the ADE. NAPSG staff provided clarification and advice of using certain symbols.
   d. If the symbol is classified as unknown, expert assistance was required to provide advice and clarification.

After this stage, the symbols were filtered again by determining whether the symbols are related to public safety or not. The final filtering was undertaken with the assistance of NAPSG. NAPSG also provided feedback on which symbols are the most common related to indoor public safety.

5.2. Step 2. Identifying the properties of public safety features

NOTE

Summary

This section explains how public safety features and their properties for ADE classes were identified from the NAPSG symbology.

After discussions with experts of NAPSG, it was concluded that there are several aspects for the public safety features that are critical in the development of the ADE. This includes the type of a feature, the floor it is located on, the location of the feature in a given floor, a description related to the feature, and ability to provide extra information such as comments made by experts. While these are abstract and general aspects, in the development of the ADE, these aspects have been regarded as the benchmark for compliance to the NAPSG requirements. The localization aspect (floor level and location) are implied in CityGML but other aspects differ based on the feature type.

For example, for features in the category of opening, e.g., “Fire Escape”, properties such as size, material, swing, direction, and lock type were identified. It was also discovered that there are features in the same category with a simple variation. In this circumstance, a Boolean property was assigned to the features. Examples of this include “Fire Pump, with Drive” and “Fire Pump, without

Drive.” An attribute was created using “Drive” as a Boolean. There were also public safety features with types and subtypes. For example, Detector has types (Heat, Gas, Smoke). Without listing all of the subtypes for each type, examples include:

- Heat
  - Flame
  - Flow
  - Gas
- Smoke
  - Beam
  - Ionization
  - Photoelectric

5.3. Step 3. Analysing CityGML ADEs

NOTE

Summary

This section explains what options are available for the development of CityGML ADEs.

The ADE is a built-in mechanism of CityGML to augment its data model with additional concepts required by particular use cases. ADEs can be modeled directly in the XML schema or can be generated by extending the UML model of CityGML with application specific information and later deriving the XML schema from it. The CityGML 2.0 standard specifies rules and guidelines for modeling ADEs in the XML schemas.

In general there are two approaches to modeling an ADE in XSD and UML. Every CityGML feature class has a GML ‘hook’ of the form `_GenericApplicationPropertyOf<Featuretypename>` in its XML schema definition which allows attachment of additional attributes by ADEs. Another approach for modeling an ADE is by extending the CityGML model with new classes for new feature types as subclasses of the existing CityGML classes. In this case, the hook mechanism is not used because a new class is added and not only properties to an existing CityGML class. The subclasses are also marked using the same stereotype («featureType») as the CityGML classes and are not suppressed in the XML Schema. The Energy ADE.[2], Noise ADE.[3], and Robotics ADE.[4] were used as samples to create the Public Safety ADE. The sample ADEs contain XSD and UML files. These were analyzed to create the structure of the Public Safety ADE.

5.4. Step 4. Identifying the relationships with existing CityGML classes

NOTE

Summary

This section demonstrates how relationships are established with the NAPSG symbology and CityGML classes.

In developing the ADE, there were four classes of CityGML identified for extension or reference to include public safety requirements. Subsequently, seven new classes were created to cater to requirements. A summary of the classes is provided below:

Extended or Referenced current classes included in CityGML:

- IntBuildingInstallation
- Opening
- Door
- Room

Newly Created Classes for the purpose of the ADE development:

- PublicSafetyIntBuildingInstallation
- PublicSafetyAlarm
- PublicSafetyRoom
- PublicSafetyDoor
- Hatch
- PublicSafetyWindow
- PublicSafetyDetectors, etc.

As is evident, the majority of the features were related to `IntBuildingInstallation` as most of the public safety related features in the current model focused more on internal building installations specifically related to public safety.

Features from the NAPSG symbology are subdivided into 6 main legend components, which in the `PublicSafetyIntBuildingInstallation` are used as properties to connect to zero- to-many instances of separate classes for `PublicSafetyAlarms, PublicSafetyDetectors,` etc. This allows Feature Types to have their own separate properties, which specifies public safety related properties to each internal installation. The legend components are implemented in the ADE using an enumeration code list.

Furthermore, the PublicSafety ADE identifies `Room` in CityGML as a extension to enable public safety-specific properties. This is done by introducing `PublicSafetyRoom. Room` can be specified based on its purpose such as `medical` and `security` using an enumeration code list based on the NAPSG symbology.

`Door` and `Window` are used to extend `PublicSafetyDoor` and `PublicSafetyWindow` to accommodate the CityGML properties related to public safety for a door and window.

`Hatch` is a new FeatureType introduced in the PublicSafety ADE. It is enabled as a new type of opening in the floor or ceiling. `Hatch` is not a window nor a door. However, a `hatch` has similar properties to a window or a door. `Hatch` is connected the `AbstractOpening` class similar to the `Door` and `Window` in CityGML.

5.5. Step 5. Implementation and validation

NOTE

Summary

This section describes the process by which ADE was implemented and validated.

In implementing the ADE by building on the CityGML 2.0 UML expression, a UML diagram for the ADE was developed using Enterprise Architect by Sparx System [https://sparxsystems.com.au/enterprise-architect/]. The UML diagram was then exported to an XSD file. Several Read and Write features of FME by Safe Software [https://www.safe.com] were utilized to analyze if the XSD file was valid. A custom CityGML file was programmed to test specific ADE classes. For example, the ADE class `PublicSafetyDoor` was used with a `FireEscape` property to test if the ADE was implemented properly with CityGML. A simple flat wall surface with an opening and a door was created in CityGML to implement the `PublicSafetyDoor` ADE class.

Issues found when implementing were caused by the sequence of the ADE tags and the XML headers not importing into the XSD file properly. It was found that the XML attribute, XML Namespaces `xmlns:prefix=URI` was required to import XSD files. The required URI must be valid by the CityGML reader. It was discovered that any URI can be used, as the URI has no effect on the XSD and ADE classes. The URI `http://www.citygml.org/ade/publicsafety/2.0` was used for the public safety namespace (`xmlns:ps="http://www.citygml.org/ade/publicsafety/2.0"`). This URI is not an active URL. It is only a reference for the Public Safety ADE.

[2] Energy ADE: http://www.citygmlwiki.org/index.php/CityGML_Energy_ADE

[3] Noise ADE: http://schemas.opengis.net/citygml/examples/2.0/ade/noise-ade/

[4] Robotics ADE: http://schemas.opengis.net/citygml/examples/2.0/ade/robotics-ade/

Chapter 6. XSD and UML Guide

NOTE	*Overview* This section explains the XSD file line by line, as well as the XSD diagram. It is based on the ADE-PublicSafety-ver03.xsd file.

6.1. XML

6.1.1. Header, Namespaces and Schemas

Header section of the PublicSafety XSD file

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2018 (https://www.liquid-technologies.com)-->
<xs:schema xmlns:ps="http://www.citygml.org/ade/publicsafety/2.0"
           xmlns:bldg="http://www.opengis.net/citygml/building/2.0"
           xmlns:core="http://www.opengis.net/citygml/2.0"
           xmlns:gml="http://www.opengis.net/gml"
           xmlns:gen="http://www.opengis.net/citygml/generics/2.0"
           attributeFormDefault="unqualified" elementFormDefault="qualified"
targetNamespace="http://www.citygml.org/ade/publicsafety/2.0" xmlns:xs=
"http://www.w3.org/2001/XMLSchema">
    <xs:import schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"
namespace="http://www.opengis.net/gml" />
    <xs:import schemaLocation="http://schemas.opengis.net/citygml/2.0/cityGMLBase.xsd"
namespace="http://www.opengis.net/citygml/2.0" />
    <xs:import schemaLocation=
"http://schemas.opengis.net/citygml/building/2.0/building.xsd" namespace=
"http://www.opengis.net/citygml/building/2.0" />
    <xs:import schemaLocation=
"http://schemas.opengis.net/citygml/generics/2.0/generics.xsd" namespace=
"http://www.opengis.net/citygml/generics/2.0" />
```

`<?xml version="1.0" encoding="utf-8" ?>` is the standard header for XML.

```
<xs:schema xmlns:ps="http://www.citygml.org/ade/publicsafety/2.0"
           xmlns:bldg="http://www.opengis.net/citygml/building/2.0"
           xmlns:core="http://www.opengis.net/citygml/2.0"
           xmlns:gml="http://www.opengis.net/gml"
           xmlns:gen="http://www.opengis.net/citygml/generics/2.0"
           attributeFormDefault="unqualified" elementFormDefault="qualified"
targetNamespace="http://www.citygml.org/ade/publicsafety/2.0" xmlns:xs=
"http://www.w3.org/2001/XMLSchema">
```

The section above provides the XSD file with XML namespaces. XML namespaces are imported using a URI. This is done in the structure `xmlns:prefix="URI"`.

For example:

```
xmlns:ps="http://www.citygml.org/ade/publicsafety/2.0"
```

The “ps” is the prefix. This prefix is designated for the PublicSafety ADE. The prefix allows users to refer to the PublicSafety ADE classes when importing into a CityGML by using the `ps` prefix. For example users can reference the hatch class in a CityGML file with `ps:hatch`.

The other namespaces imported are standard XML and CityGML namespaces.

```
xmlns:xs=“http://www.w3.org/2001/XMLSchema”
```

The line above imports the standard namespace for XSD files.

```
xmlns:bldg=“http://www.opengis.net/citygml/building/2.0”
```

The line above imports the building class from CityGML. The building class references all aspects relating to buildings in CityGML. This includes building parts and installations.

```
xmlns:core="http://www.opengis.net/citygml/2.0"
```

The line above refers to the CityGML core namespace, which features all the main components of CityGML. This is required when using CityGML.

```
xmlns:gml="http://www.opengis.net/gml"
```

The line above imports the GML namespace.

The XML attribute `targetNamespace` states that the elements defined in the XSD, are from the `“http://www.citygml.org/ade/publicsafety/2.0”` namespace.

`elementFormDefault=“qualified”` states that elements that are used in a XML instance document from this XSD file are required to be namespace qualified. This means PublicSafety ADE elements must be used with the namespace prefix. In this case `ps`. Likewise, `attributeFormDefault=“unqualified”` states that attributes do not require to be namespace qualified.

```
<xs:import schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"
namespace="http://www.opengis.net/gml" />
    <xs:import schemaLocation="http://schemas.opengis.net/citygml/2.0/cityGMLBase.xsd"
namespace="http://www.opengis.net/citygml/2.0" />
    <xs:import schemaLocation=
"http://schemas.opengis.net/citygml/building/2.0/building.xsd" namespace=
"http://www.opengis.net/citygml/building/2.0" />
    <xs:import schemaLocation=
"http://schemas.opengis.net/citygml/generics/2.0/generics.xsd" namespace=
"http://www.opengis.net/citygml/generics/2.0" />
```

The lines above import different XSD files with their respective target namespaces. The XML attribute `schemaLocation` specifies the URI for the XSD. Whereas the attribute `namespace` imports the URI of the namespace.

6.1.2. Feature Type Explanation

NOTE	*Note* This section only goes over the PublicSafetyDoor lines of the XSD, as PublicSafetyRoom and PublicSafetyIntBuildingInstallation are very similar. This is further explained in the UML diagram section.

Overview of the PublicSafetyDoor Section

```
<xs:element name="PublicSafetyDoorProperty" type="ps:PublicSafetyDoorPropertyType"
            substitutionGroup="bldg:_GenericApplicationPropertyOfDoor" />

<xs:complexType name="PublicSafetyDoorPropertyType">
    <xs:sequence minOccurs="0">
        <xs:element ref="ps:PublicSafetyDoor" />
    </xs:sequence>
    <xs:attributeGroup ref="gml:AssociationAttributeGroup" />
</xs:complexType>

<xs:complexType name="PublicSafetyDoorType">
    <xs:complexContent>
        <xs:extension base="bldg:AbstractOpeningType">
            <xs:sequence>
                <xs:element name="doorHandling" type="ps:doorHandling" minOccurs="1"
maxOccurs="1" />
                <xs:element name="doorMaterial" type="xs:string" minOccurs="1"
maxOccurs="1" />
                <xs:element name="doorSwing" type="ps:doorSwing" minOccurs="1"
maxOccurs="1" />
                <xs:element name="fireDoor" type="xs:boolean" minOccurs="0" maxOccurs
="1" />
                <xs:element name="lockType" type="xs:string" minOccurs="1" maxOccurs=
"1" />
            </xs:sequence>
```

```
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<xs:element name="PublicSafetyDoor" type="ps:PublicSafetyDoorType" substitutionGroup=
"bldg:_Opening" />

<xs:simpleType name="doorHandling">
    <xs:annotation>
        <xs:documentation>This is to decided weather the door is opening from right or
left side</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
        <xs:enumeration value="Left" />
        <xs:enumeration value="Right" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="doorSwing">
    <xs:annotation>
        <xs:documentation>This is to decided weather the door is opening to inward or
outward</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
        <xs:enumeration value="Inward" />
        <xs:enumeration value="Outward" />
    </xs:restriction>
</xs:simpleType>
```

The whole code section above relates to creating a Public Safety Door feature type.

PublicSafetyDoor Section 1

```
<xs:element name="PublicSafetyDoorProperty" type="ps:PublicSafetyDoorPropertyType"
            substitutionGroup="bldg:_GenericApplicationPropertyOfDoor" />
```

The lines above creates an XML element that contains PublicSafetyDoor feature type properties. The XML attribute `name` provides the XML element with a name. The XML attribute `type` specifies whether the element is a built-in XML data type, or an XML element defined by simpleType/complexType. substitutionGroup provides another element that allows the current element to substitute it. In this case, the element `PublicSafetyDoorProperty` can substitute the element `bldg:_GenericApplicationPropertyOfDoor` from CityGML.

PublicSafetyDoor Section 2

```
<xs:complexType name="PublicSafetyDoorPropertyType">
   <xs:sequence minOccurs="0">
       <xs:element ref="ps:PublicSafetyDoor" />
   </xs:sequence>
   <xs:attributeGroup ref="gml:AssociationAttributeGroup" />
</xs:complexType>
```

The code section above provides a link to XML element created in PublicSafetyDoor Section 1. This section creates a `complexType` of `PublicSafetyDoorPropertyType`. The XML element `xs:sequence` specifies that its child elements are required to be used in that sequence. The XML attribute `minOccurs` provides the minimum amount of times the sequence can occur. The XML element `xs:element ref="ps:PublicSafetyDoor"` makes a reference to `ps:PublicSafetyDoor`. This provides the relationship between the `PublicSafetyDoor` element the code in PublicSafetyDoor Section 3 as a child element of `PublicSafetyDoorPropertyType`. The XML element `<xs:attributeGroup ref="gml:AssociationAttributeGroup" />` with an XML attribute `ref` provides a reference to an attribute group. In this case, the `gml:AssociationAttributeGroup`.

PublicSafetyDoor Section 3

```
<xs:element name="PublicSafetyDoor" type="ps:PublicSafetyDoorType" substitutionGroup=
"bldg:_Opening" />
```

PublicSafetyDoor Section 4

```
<xs:complexType name="PublicSafetyDoorType">
    <xs:complexContent>
        <xs:extension base="bldg:AbstractOpeningType">
            <xs:sequence>
                <xs:element name="doorHandling" type="ps:doorHandling" minOccurs="1"
maxOccurs="1" />
                <xs:element name="doorMaterial" type="xs:string" minOccurs="1"
maxOccurs="1" />
                <xs:element name="doorSwing" type="ps:doorSwing" minOccurs="1"
maxOccurs="1" />
                <xs:element name="fireDoor" type="xs:boolean" minOccurs="0" maxOccurs
="1" />
                <xs:element name="lockType" type="xs:string" minOccurs="1" maxOccurs=
"1" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

PublicSafetyDoor Section 4 defines feature type elements that are to be used as attributes for `PublicSafetyDoor`. `<xs:extension base="bldg:AbstractOpeningType">` states that the `PublicSafetyDoorType` is extended from the CityGML building module with the tag,

bldg:AbstractOpeningType. The XML elements with xs:element define properties for PublicSafetyDoor. The elements with the name attributes doorMaterial, fireDoor and lockType use built-in XML types, however doorHandling and doorSwing use declared simpleType elements from the PublicSafety XSD file. These simpleType elements are defined in line 42 and 52. The XML attributes minOccurs and maxOccurs designate the minimum and maximum amount of times the PublicSafetyDoor property can occur. With both XML attributes equaling to 1, this makes these PublicSafetyDoor properties a requirement.

The line in PublicSafety Section 3 instantiates the element PublicSafetyDoor.

PublicSafetyDoor Section 5

```
<xs:simpleType name="doorHandling">
    <xs:annotation>
        <xs:documentation>This is to decided weather the door is opening from right or
left side</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
        <xs:enumeration value="Left" />
        <xs:enumeration value="Right" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="doorSwing">
    <xs:annotation>
        <xs:documentation>This is to decided weather the door is opening to inward or
outward</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
        <xs:enumeration value="Inward" />
        <xs:enumeration value="Outward" />
    </xs:restriction>
</xs:simpleType>
```

PublicSafetyDoor Section 5 creates a simpleType element for the type XML attributes in PublicSafetyDoor Section 4. The xs:annotation elements create annotations for user feedback. The xs:restriction element adds a restriction that only allows the user to select the values defined by the xs:enumeration elements.

The process of PublicSafetyDoor feature type as shown in the Overview of the PublicSafetyDoor Section similarly repeats the same structure with PublicSafetyIntBuildingInstallation and PublicSafetyRoom feature types. PublicSafetyHatch however has a slightly different structure. It does not require as much referencing to the CityGML modules. Therefore, the PublicSafetyHatch was more simple to implement. The UML diagrams in the next section below explains this.

PublicSafetyHazard Section 1

```
<xs:complexType name="PublicSafetyIntBuildingInstallationType">
   <xs:complexContent>
       <xs:extension base="core:AbstractCityObjectType">
           <xs:sequence>
               <xs:element name="PSInstallationLegend" type="ps:PsLegend" minOccurs="
0" maxOccurs="1" />
               <xs:element name="installedAlarm" type=
"ps:PublicSafetyAlarmPropertyType" minOccurs="0" maxOccurs="unbounded" />
               <xs:element name="knownHazards" type="
ps:PublicSafetyHazardPropertyType" minOccurs="0" maxOccurs="unbounded" />
           </xs:sequence>
       </xs:extension>
   </xs:complexContent>
</xs:complexType>

<xs:complexType name="PublicSafetyHazardPropertyType">
   <xs:sequence>
       <xs:element name="PublicSafetyHazard" type="ps:PublicSafetyHazardType"/>
   </xs:sequence>
   <xs:attributeGroup ref="gml:AssociationAttributeGroup"/>
</xs:complexType>

<xs:complexType name="PublicSafetyHazardType">
   <xs:complexContent>
       <xs:extension base="gml:AbstractFeatureType">
           <xs:sequence>
               <xs:element name="hazardLocation" type="xs:string" minOccurs="0"
maxOccurs="1" />
               <xs:element name="hazardDescription" type="xs:string" minOccurs="0"
maxOccurs="1" />
           </xs:sequence>
       </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

Public Safety Hazards is a feature type created for known hazards in a building. This feature type has elements such as description and location for the hazard.

The PublicSafetyHazard feature type is under the `PublicSafetyIntBuildingInstallation` feature type.

6.1.3. UML Diagrams

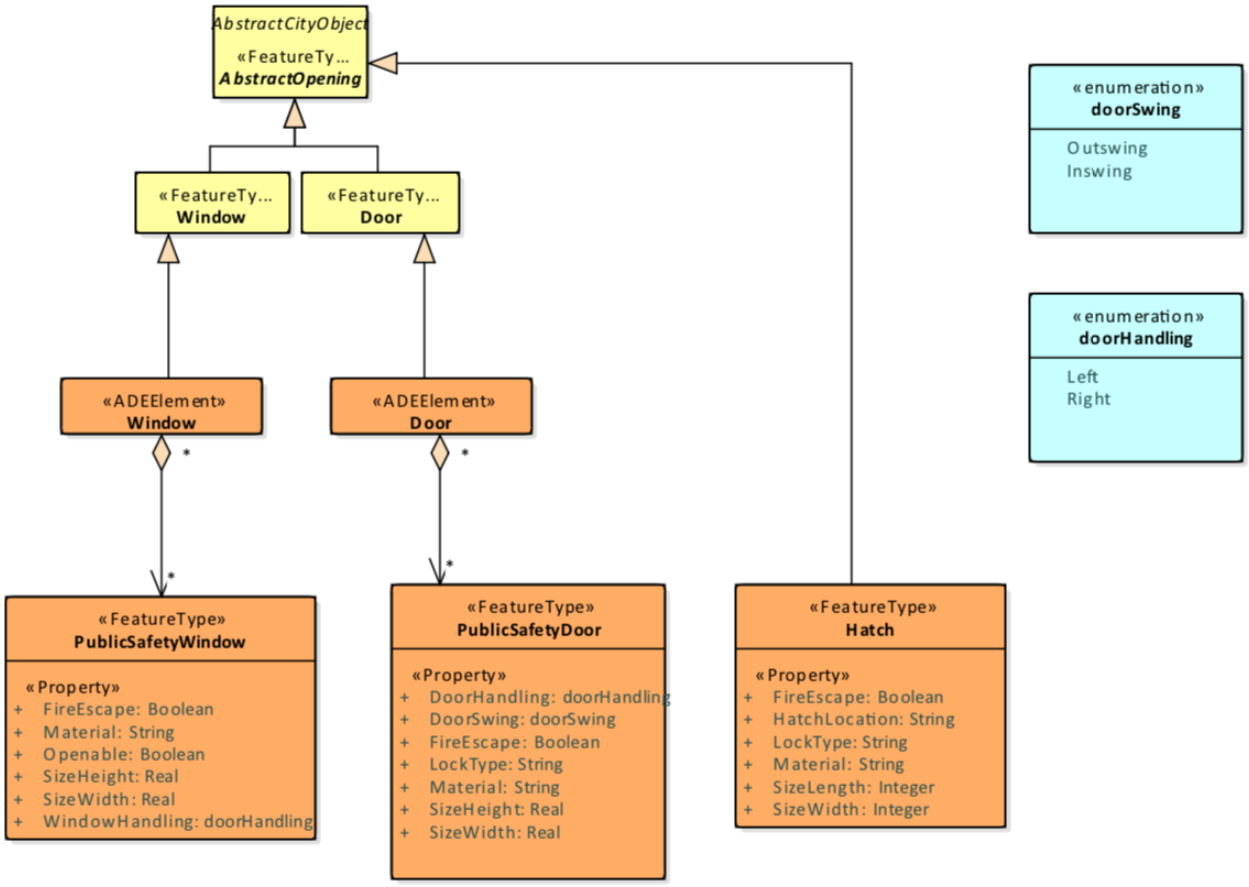

Figure 1. UML Diagram for ADE feature types based on AbstractOpening

As shown in the Figure 1, `PublicSafetyWindow` and `PublicSafetyDoor` both are extended from `ADEElements` `Window` and `Door` which are extended from the CityGML window and door feature type. CityGML window and door feature types are extended from the CityGML `AbstractOpening` feature type. Since there are more extensions for these two feature types, there was more lines of code to form these relationships. However, the `Hatch` feature type simply is extended from `AbstractOpening` in CityGML. Therefore it requires less lines of code in the XSD file.

The UML diagram also shows all the properties for the Public Safety ADE feature types. As shown, most properties use built-in XML data types. However, enumerations were created for the `doorSwing` and `doorHandling` properties.

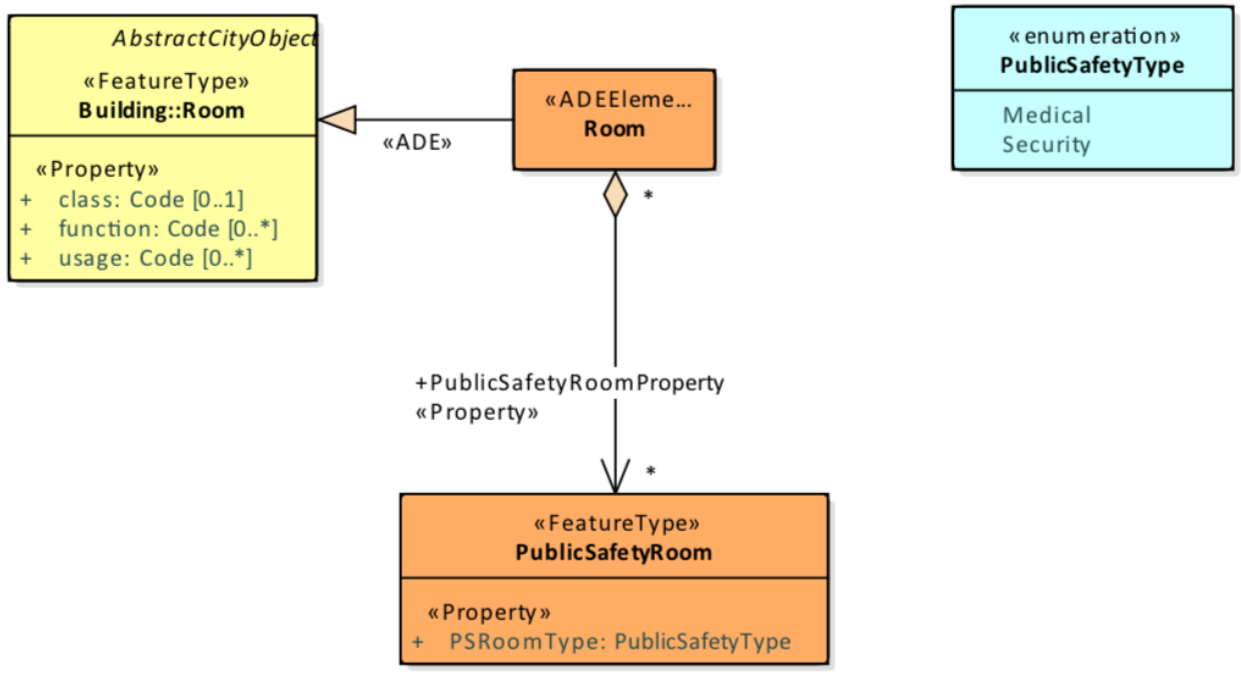

Figure 2. UML Diagram for ADE feature type based on bldg:room

Like PublicSafetyDoor and PublicSafetyWindow, PublicSafetyRoom follows the same relationship structure with a small difference of extending from CityGML bldg:room as shown in Figure 2.

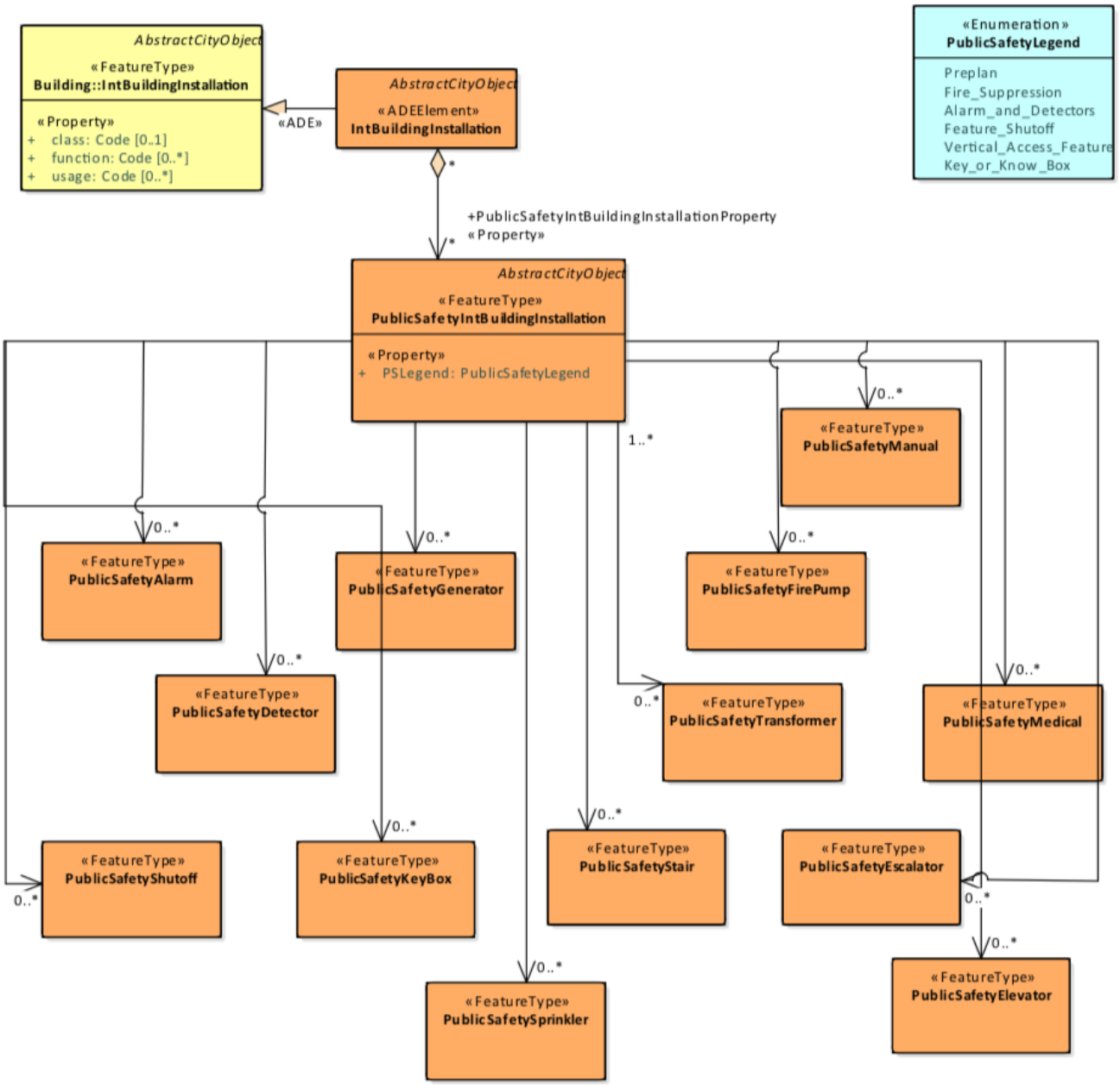

Figure 3. UML Diagram for ADE feature type based on bldg:IntBuildingInstallation

As shown in Figure 3, the PublicSafetyIntBuildingInstallation feature type is slightly different. It has many sub feature types. It also contains a property which defines the category of the sub feature type. The relationship between PublicSafetyIntBuildingInstallation and the sub feature types is 0 to many.

Chapter 7. Discussion

With developing the ADE, there was no comprehensive, dedicated guidelines or instructions that could be found. There was some knowledge in academic papers and a high-level guideline in the OGC CityGML specification. However, this level of knowledge was scattered and not well organized. This gap could be an impediment for non-spatial disciplines for developing ADEs and subsequently a barrier to the widespread adoption of CityGML for other applications. □□

However, there are several ADEs that have been developed over many years. The ADEs are a valuable resource to investigate options, identify best practices, and develop alternatives. As mentioned, the ADEs are generally not accompanied by a great deal of documentation. A significant effort goes into the interpretation and understanding of the logic behind an ADE.□□

The Public Safety ADE has been developed based on a list of symbology guided by a public safety expert advice. There is still room for improvement by considering and developing use cases for the ADE. Additionally, the ADE has only been tested using simulated data where public safety elements were manually added to an existing dataset. The validity of the ADE was tested by ensuring the simulated dataset was enriched by the ADE elements can be parsed, queried, and visualized. The assessment of the ADE can be taken to the next level in which a dataset with public safety features embedded is transformed to a CityGML file format that is extended by the ADE.

Chapter 8. Conclusions

8.1. Conclusions

This ER provided the documentation on the design, development and implementation of a CityGML ADE for public safety applications. The ADE specifically focuses on indoor public safety and features that are critical for the preparatory stages of emergency response.

In developing the ER, NAPSG was used as an input to gather the requirements for the development of the ADE. In gathering the requirements, the methodology involved extracting data elements from a set of symbols representing public safety features. Much of the features types and properties interpreted through interaction with public safety experts and their interpretation of the symbols and implied application of them. Building on the CityGML 2.0 UML diagram, the ADE was implemented in which several existing elements either referenced or extended and several new features were created. The ADE was evaluated by successfully applying it in some sample CityGML files.

8.2. Future Work

In the next step of the Indoor Mapping and Modeling Pilot project, it is required to use the ADE to examine further if the ADE meets the requirement of the use cases in navigation preplanning for public safety. For example, it is required to examine if there is a need to add more features/properties relevant to navigation scenario or if the mapping from CityGML ADE to IndoorGML can be done without losing public safety elements in the transformation of the data. It is also important to note that while there are many CityGML ADEs available publicly, there is a no extensive official guide for creating ADEs, making a less structured and more subjective task. It is recommended to develop such a guide for future work.

Appendix A: Revision History

Table 1. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
January 25, 2019	S. Chau	.3	all	draft version
January 16, 2019	S. Chau	.2	all	draft version
December 18, 2018	M. Kalantari	.1	all	draft version