Streamer
European research on energy-efficient healthcare districts

D8.8

Recommendation to buildingSMART, OGC, W3C on open standards improvement

Deliverable Report: D8.8, Final version

Streamer - Optimised design methodologies for energy-efficient buildings integrated in the neighbourhood energy systems.
The Streamer project is co-financed by the European Commission under the seventh research framework programme FP7.EeB.NMP.2013-5 ; GA No. 608739)

D8.8

Recommendation to buildingSMART, OGC, WC3 on open standards improvement

Main author Karl-Heinz Häfele, Joachim Benner (KIT)
Co-authors Nick Nisbet (AEC), Matthias Weise (AEC), Andreas Geiger (KIT)
Dissemination R

Document history

Version	Date	Status	Produced/Reviewed by	Comments
	2016/10/14	Sketch	Karl-Heinz Häfele (KIT)	
18	2017/06/27	Final draft	Karl-Heinz Häfele, Joachim Benner (KIT)	
19	2017/07/16	Reviewed	CSTB and TNO	
20	2017/07/20	Accepted	Marc Bourdeau (TC)	
21	2017/08/17	Approved	Freek Bomhof (PC)	

Colophon

The Streamer project is co-financed by the European Commission under the seventh research framework programme with contract No.: . 608739 - FP7-2013-NMP-ENV-EeB. The information in this publication does not necessarily represent the view of the European Commission. The European Commission shall not in any way be liable or responsible for the use of any such knowledge, information or data, or of the consequences thereof.

Publishable executive summary

All over the STREAMER project, data exchange via standardised data models of the Building Information Modelling (BIM) and Geographic Information System (GIS) areas played an important role. In many deliverables, it became obvious that the existing versions of the standards still have gaps and deficiencies. The corresponding experiences are documented in this report, and in most cases recommendations for specific improvements are stated. Furthermore, product spectrum, goals and organisation structure of the two most important standardisation bodies in the BIM area (buildingSMART International. bSI) and the GIS area (Open Geospatial Consortium, OGC) are described.

In the BIM area, the three standards mainly used in STREAMER are "Industry Foundation Classes" (IFC), "Model View Definition XML" (mvdXML) and "buildingSMART Data Dictionary" (bsDD), formerly called "International Framework for Dictionaries2 (IFD). The experiences gained in STREAMER resulted in altogether 24 different detailed recommendations, most of them (14) are related with the base standard IFC. Among these are proposals for improving the documentation, new implementer agreements, new or modified IFC entities, as well as suggestions for a number of new or extended property sets related with STREAMER relevant building properties. Part of them have already been submitted to the corresponding standardisation groups.

The mvdXML related recommendations (altogether 9) are related with the existing version 1.1 and the proposed next version 2.0 of the standard. They aim at improving the formal language to specify checking rules, and to enhance the readability of these rules. For bsDD, only the general recommendation to host attribute synonyms, including proprietary attributes from applications like Revit or SBEM is stated.

For representing GIS related data, mainly the CityGML standard was used in the STREAMER project. The report lists 14 topics where the base standard needs improvements or application specific extensions. This includes general problems like improving quality of the documentation, revising the CityGML module structure and naming concept, and generalising the Level of Detail (LoD) concept. Furthermore, specific suggestions for extending the CityGML functionality to represent the spatial structure of cities, city quarters and buildings, and utility networks are stated.

During the first years of the STREAMER project time it turned out that Semantic Web technologies like OWL are not such extensively used as originally expected. In consequence, no recommendations for the corresponding standardisation organisation W3C could be derived.

List of acronyms and abbreviations

- ADE: Application Domain Extension
- AEC: Architecture, Engineering and Construction
- BCF: BIM Collaboration Format
- BIM: Building Information Modelling
- BS: British Standards
- bSI: buildingSMART International
- bSDD: buildingSMART Data Dictionary
- CityGML: City Geography Markup Language
- CRS: coordinate reference system
- EPD: Environmental Product Declaration
- GIS: Geographic Information System
- GML: Geography Markup Language
- IFC: Industry Foundation Classes
- IFD: International Framework for Dictionaries
- ifcOWL: Web Ontology Language (OWL) representation of the IFC schema
- ILCD: International Reference Life Cycle Data System
- ISG: Implementation Support Group (buildingSMART)
- ISO: International Organization for Standardization
- KIT: Karlsruhe Institute of Technology
- LCA: Life Cycle Assessment
- LCDN: Life Cycle Data Network
- LoD: Level of Detail
- MSG: Model Support Group (buildingSMART)
- OGC: Open Geospatial Consortium
- OWL: Web Ontology Language
- PLM: Product Lifecycle Management
- PMO: Product Modelling Ontology
- RDF: Resource Description Framework
- SC: Subcommittee
- SIG3D: Special Interest Group 3D of German Spatial Data Infrastructure
- STREAMER: Semantics-driven Design through Geo and Building Information Modelling for Energy-efficient Buildings Integrated in Mixed-use Healthcare Districts
- SWG: Standard Working Group (OGC)
- TC: Technical Committee
- WG: Working Group
- WWW: World Wide Web
- XML: Extensible Markup Language

Definitions

BIM – Building Information Modeling (BIM) is a digital representation of physical and functional characteristics of a facility. A BIM is a shared knowledge resource for information about a facility forming a reliable basis for decisions during its life cycle; defined as existing from earliest conception to demolition [BIM2016].

BIM – "use of a shared digital representation of a built object (including buildings, bridges, roads, process plants, etc.) to facilitate design, construction and operation processes to form a reliable basis for decisions

Note 1 to entry: The acronym BIM also stands for the shared digital representation of the physical and functional characteristics of any construction works" [ISO29481-1].

De facto standard – A de facto standard is a custom, convention, product, or system that has achieved a dominant position by public acceptance or market forces (such as early entrance to the market). De facto is a Latin phrase that means in fact (literally by or from fact) in the sense of "in practice but not necessarily ordained by law" or "in practice or actuality, but not officially established", as opposed to de jure [Defacto2016].

Entity – A class of information defined by common properties [ISO10303-11].

Feature – A feature is an abstraction of real world phenomena [ISO19101-1].

Level of Detail (CityGML) – In computer graphics, the level of detail describes the complexity of the representation of a 3D object. Beside the geometrical complexity of the object, CityGML introduces different semantic levels for each level of detail [CityGML2012].

Life cycle assessment – Compilation and evaluation of the inputs, outputs and the potential environmental impacts of a product system throughout its life cycle [ISO14040].

Model View Definition (MVD): – "computer-interpretable definition of an exchange requirement, specifically bound to one or more particular standard information schemas

Note 1 to entry: A model view definition (MVD) is also referred to as a view definition, a subset (of a schema) and a conformance class (CC) especially in ISO 10303" [ISO29481-1].

openBIM – openBIM is a universal approach to the collaborative design, realization and operation of buildings based on open standards and workflows. openBIM is an initiative of buildingSMART and several leading software vendors using the open buildingSMART Data Model [bSIopenBIM2017].

Standard – A standard is a document that provides requirements, specifications, guidelines or characteristics that can be used consistently to ensure that materials, products, processes and services are fit for their purpose [ISOStandard2016].

Standardisation – Standardization or standardisation is the process of implementing and developing technical standards based on the consensus of different parties that include firms, users, interest groups, standards organizations and governments [Standardisation2016].

Typical meteorological year – "A typical meteorological year (TMY) is a collation of selected weather data for a specific location, generated from a data bank much longer than a year in duration. It is specially selected so that it presents the range of weather phenomena for the location in question, while still giving annual averages that are consistent with the long-term averages for the location in question" [TMY2017].

Contents

1. Introduction

For complex planning processes as they have been studied in STREAMER, many data from different application domains need to be processed and integrated by various applications. Due to the diversity of software tools in use, only data available in open and standardised data formats can be processed with reasonable effort.
Thus, standardisation of data exchange formats is a crucial factor for the design processes regarded in STREAMER, and task 8.4 "BIM, GIS, Semantic Web open-standardisation" is devoted to collect and document the experiences of the STREAMER project on deficiencies and gaps in existing standards.
The title of the corresponding deliverable D 8.8 explicitly mentions three standardisation organisations:

- **buildingSMART International Ltd.,** responsible for various Building Information Modelling (BIM) standards;
- **Open Geospatial Consortium Inc.** (OGC), responsible for standards in the area of Geographical Information System (GIS), and
- **W3C Consortium**, managing a number of Semantic Web standards.

During the first years of the project time, it turned out that semantic web technologies like RDF or OWL are not thus important for STREAMER as expected in the STREAMER proposal. In consequence, no recommendations to the W3C consortium regarding semantic web standards can be formulated. This report therefore concentrates on the STREAMER relevant standards in the BIM and GIS area, and lists a number of major deficiencies, together with a lot of recommendation for potential improvements. Furthermore, some basic information on the mentioned standardisation organisations, including basic information on their mission, organisation structure, processes, and a comprehensive list of standards, are given.

2. Standardisation Organisations

As STREAMER is covering energy-efficient buildings including the building's neighbourhood and energy system, several standards, developed by different organisations, are used to establish interoperability. For STREAMER relevant data, the most important standardisation organisations are:

- **buildingSMART International Ltd.** – home of openBIM, and
- **Open Geospatial Consortium Inc.** – Making location count

In this section, some basic information on these organisations, their processes and the relevant standards will be given. This information is essential for everyone who wants to influence and support the advancement of existing standards.

2.1 buildingSMART International (bSI)

"buildingSMART is the worldwide authority driving the transformation of the built asset economy through creation & adoption of open, international standards" [bSIabout2016]. buildingSMART focuses on standardising processes, workflows and procedures for Building Information Modelling. The standardisation processes are supported by the buildingSMART's competences in:

- Methodologies to describe processes
- Data Modelling to transport information
- Mapping terms
- Methodology and technology to transform process requirements into technical requirements

Standards

Alongside the well-known IFC (Industry Foundation Classes) product data model for buildings, buildingSMART offers standards for the collaboration in the design process (BCF), a framework for dictionaries (IFD) and a formal description for IFC model view definitions (mvdXML). There are several model view definitions (including the Design Transfer View and Reference View), which are also buildingSMART standards and which are the base for the IFC certification. With the Information Delivery Manual (IDM), a standard is available, which describes the connection between processes of a specific project and the data necessary to perform these processes. Table 1 gives an overview of the STREAMER relevant buildingSMART standards.

Standard	Long Name	Scope	Remarks
IDM	Information Delivery Manual	Processes	ISO 29481-1 [ISO29481-1], ISO 29481-2 [ISO29481-2]
IFC 4	Industry Foundation Classes Version 4	Building	ISO standard (ISO 16739:2013) [ISO16739]

BCF XML	BIM Collaboration Format XML	Collaboration	
BCF API	BIM Collaboration Format REST API	Web Service	
bSDD (IFD)	International Framework for Dictionaries	Product data Classifications	
mvdXML	Model View Definition XML	Formal description of Model View Definition	
Design Transfer View	IFC 4 Design Transfer View	Model View Definition to hand over models to perform in next work flows, allowing modifications of its content	
Reference View	IFC 4 Reference View	Model View Definition to hand over models for downstream applications, which usually don't perform modifications	
Infrastructure Alignment	IFC 4 Infrastructure Alignment	Model View Definition to hand over 3D and 2D alignment information for spatial location of infrastructure assets	

Table 1: List of STREAMER relevant buildingSMART standards

Organisation

buildingSMART is subdivided in chapters and members. Chapters are local organisations in specific countries or regions, which promote and implement openBIM. Currently there are 18 Chapters representing Australasia, Benelux, Canada, China, France, German speaking, Hong Kong, Italy, Japan, Korea, Malaysia, Nordic (Denmark, Finland & Sweden), Norway, Singapore, Spain, Switzerland, United Kingdom & Ireland and USA. All chapters are members of buildingSMART International. In addition, the membership of buildingSMART International is open to corporate entities worldwide [bSIabout2016]. The bodies of the buildingSMART International are depicted in Figure 1.

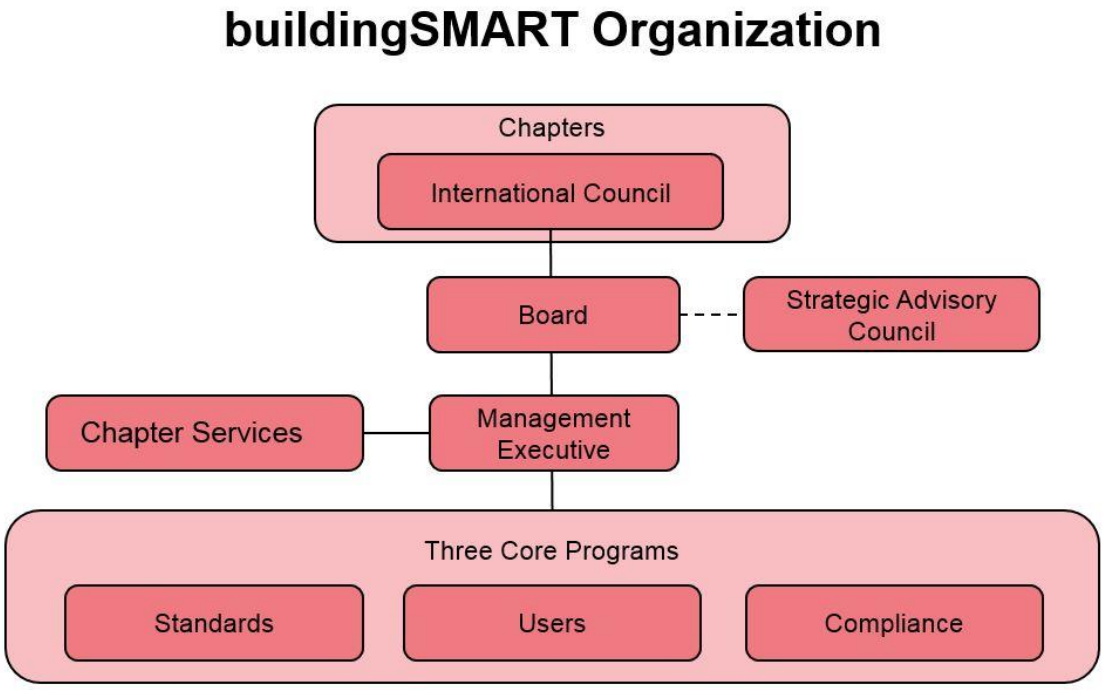

Figure 1: Organisation chart of buildingSMART International [bSIorgans2016]

The International Council, which consists of two representatives from each buildingSMART chapter, is the legal governing body of buildingSMART international. In order to get advice and feedback from the building and infrastructure industries, the Strategic Advisory Council (SAC) was established. The SAC consists of senior representatives from leading building and infrastructure industries all over the world. The Board is leading buildingSMART International and the worldwide Chapter network. It is elected by the International Council.

The Management Executives is represented by full time executives, supported by a management executive made up from members who assist in the operational leadership [bSIorgans2016].

The Three Core Programs are the basis of buildingSMART International. They are intended to support the full life cycle of standards, ranging from the requirement analysis to solution development and application to meet user needs. The Three Core Programs are:

- Standards – Open to all; formal and rigorous standards development,
- Users – Chapter outreach and coordination; chapter building,
- Compliance – Certification for software, people, companies; compliance training and testing.

Standardisation

In buildingSMART, standardisation is organised and managed by the Standards Committee. It has the executive responsibility for establishing and managing the standards program. Two of the important work groups are the Model Support Group and the Implementation Support Group.

Model Support Group (MSG)

The MSG is an international expert group of buildingSMART members. This group is responsible for the development and the maintenance of buildingSMART data model standards.

One of the main objectives of MSG is the further development of IFC like consolidation and final integration of new model specifications as well as the coordination of Model View Definitions (MVD) to define subsets of IFC for data exchange, sharing use cases, or exchange requirements. In collaboration with other teams specifications like Information Delivery Manual (IDM), openBIM Collaboration Format (BCF) and the international framework for buildingSMART Data Dictionary (bSDD, former IFD) are developed. In addition to this development tasks, the MSG also works with other buildingSMART committees and supports the Implementation Support Group (ISG), plus external standardisation groups to harmonise IFC definitions with other ISO standard deliverables [bSIMSG2017].

Implementation Support Group (ISG)

The Implementation Support Group of buildingSMART International has been established to support the implementation and certification activities for buildingSMART standards. Especially, the ISG is discussing implementation issues regarding the IFC and BCF standards. If necessary, the ISG agrees on so-called implementer agreements.

Together with the MSG, the ISG discusses the official buildingSMART model views (currently the IFC 4 Reference View [IFC4RefView2017] and the IFC 4 Design Transfer View [IFC4DTView2016]). These views are then the basis for the IFC certification, which is also be supervised be the ISG.

The membership of the ISG is open to all bSI members, which are implementing import and export functionalities for applications or which are developing downstream applications based on openBIM.

In addition, the ISG is representing the implementer interests within the buildingSMART organisation [bSIISG2017].

2.2 Open Geospatial Consortium (OGC)

The Open Geospatial Consortium (OGC) is an international organisation for development and implementation of open standards for geospatial content and services. Originated in 1994, the OGC has more than 500 members from commercial, governmental, non-profit and research organisations worldwide [OGCWikipedia2016].
The OGC mission is "to advance the development and use of international standards and supporting services that promote geospatial interoperability. To accomplish this mission, OGC serves as the global forum for the collaboration of geospatial data / solution providers and users" [OGCvision2016].

OGC follows five strategic goals [OGCvision2016]:

- Goal 1 - Provide free and openly available standards to the market that are of tangible value to members and have measurable benefits for users.
- Goal 2 - Lead worldwide in the creation and establishment of standards that enable global infrastructures for delivery and integration of geospatial content and services into business and civic processes.
- Goal 3 - Facilitate the adoption of open, spatially enabled reference architectures in enterprise environments worldwide.
- Goal 4 - Advance standards to support formation of new and innovative markets and applications for geospatial technologies.
- Goal 5 - Accelerate market assimilation of interoperability research through collaborative consortium processes.

Standards

Currently (March 2017), the list of OGC standards comprise 54 items. Table 2 depicts the subset of these standards with is relevant for STREAMER. These standards include data models / data exchange formats for geospatial content including spatially located sensors, as well as standards for web services.

Standard	Long Name	Scope
GML	Geography Markup Language	XML grammar for expressing geographical features. GML is the basis for all GML application schemata like CityGML, IndoorGML etc.
CityGML	City Geography Markup Language	Format for the storage and exchange of virtual 3D city models
IndoorGML	Indoor Geography Markup Language	Open data model and XML schema for indoor spatial information.
KML	Keyhole Markup Language (formerly)	International standard language focused on geographic visualisation, including annotation of maps and images.
Moving Features	Moving Features	Representations of movement of geographic features.

O&M	Observations and Measurements	XML schemas for observations, and for features involved in sampling when making observations.
SWE Common Data Model	Sensor Web Enablement Common Data Model	Low-level data models for exchanging sensor related data between nodes of the OGC Sensor Web Enablement (SWE) framework.
SWE Service Model	Sensor Web Enablement Service Model	Data types for common use across OGC Sensor Web Enablement (SWE) services.
SensorML	Sensor Model Language	A robust and semantically tied means of defining processes and processing components associated with the measurement and post-measurement transformation of observations.
SOS	Sensor Observation Service	Web service interface, which allows querying observations, sensor metadata, as well as representations of observed features.
SPS	Sensor Planning Service	Interfaces for queries that provide information about the capabilities of a sensor and how to task the sensor.
WCS	Web Coverage Service	Web service interface to multi-dimensional coverage data for access over the Internet.
WFS	Web Feature Service	A service that provides transactions on and access to geographic features in a manner independent of the underlying data store.
WMS	Web Map Service	A simple HTTP interface for requesting geo-registered map images from one or more distributed geospatial databases.
WMTS	Web Map Tile Service	A standard based solution to serve digital maps using predefined image tiles.
WPS	Web Processing Service	A web service that enables the execution of computing processes and the retrieval of metadata describing their purpose and functionality.

Table 2: List of possible STREAMER relevant OGC standards

Organisation

In Figure 2, the organisation structure of the OGC is depicted. The organisation has a Board of Directors, which is supported by two committees: The Strategic Member Advisory Committee (SMAC) and the Global Advisory Council (GAC). In the standards program, two other committees play a key role: The Technical Committee (TC) is responsible for all aspects of the formal consensus OGC specification process, and the Planning Committee (PC), which has ultimate responsibility for approving Technical Committee recommendations for the adoption and release of OGC standards, and for Specification Program planning.

Above all, the TC provides an open forum for professional discussion of issues and items related to the consensus development and/or evaluation and approval of standards that provide the ability to build and deploy interoperable geospatial solutions in the larger IT domain. The TC also has methodology in place to govern its composition, its meetings and meeting agendas, and voting procedures.

In order to carry out the business of the TC in a timely manner, four different types of subgroups of the TC may be formed:

- Subcommittees (SC), with a mission to provide recommendations to the TC or PC in some general area;
- Domain Working Groups (DWG), providing a forum for discussion of key interoperability requirements and issues, discussion and review of implementation specifications, and presentations on key technology areas relevant to solving geospatial interoperability issues.
- Standards Working Groups (SWG), which have specific charter of working on a candidate standard prior to approval as an OGC standard or on revising an existing OGC standard.
- Special Interest Groups (SIG), providing a forum for the discussion of business and technology interoperability requirements for given geographic region or information community.

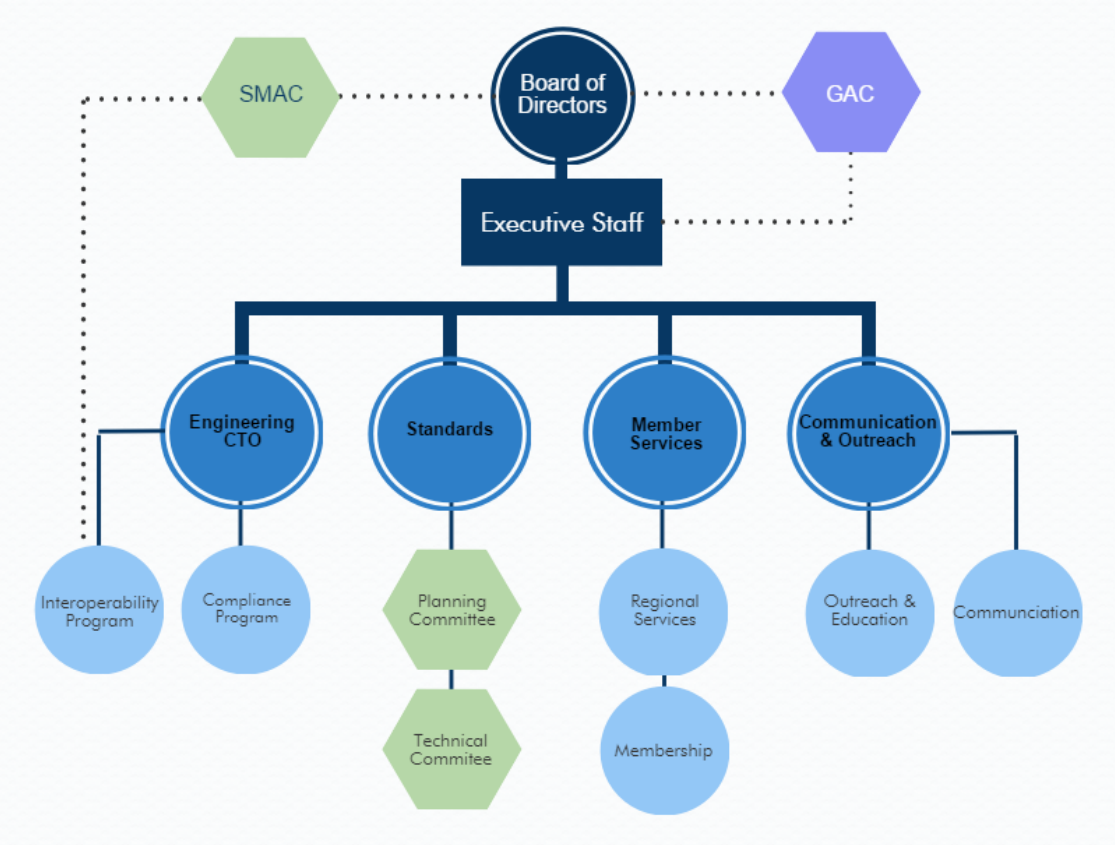

Figure 2: Organisation structure of the OGC

Standardisation of CityGML

From the OGC standards, CityGML is the most relevant in the STREAMER context. Therefore, the CityGML standardisation process and the different working groups being active in the process will be shortly described.

Mainly responsible for the standardisation of CityGML is the CityGML SWG [CityGML SWG]. Now, the SWG is technically preparing a new version CityGML 3.0. For this, 14 different technical areas have been identifies where the actual version 2.0 needs improvement and extension [CityGMLGitHub]. On German national level, the CityGML SWG is supported by the Special Interest Group 3D (SIG 3D) [SIG3D2017].

In parallel with the development of CityGML 3.0, extensions of the base standards for specific applications are being developed. These extensions use the Application Domain Extension (ADE) mechanism, which is an integral part of the CityGML standard. A CityGML ADE consists of one or more XML schema files, extending the base standard in two different ways:

- By defining new feature classes, optionally extending existing CityGML feature classes, and
- By defining new attributes or relations of existing CityGML feature classes.

For STREAMER, two of these extensions are of special interest

- **CityGML Energy ADE**: A standardised data model based on CityGML format for urban energy analyses, aiming to be a reference exchange data format between different urban modelling tools and expert databases. It extends the CityGML 2.0 standard with energy related entities and attributes necessary to perform energy analyses on urban scale.
- **CityGML Utility Networks ADE**: This extension defines a topological network model facilitating sophisticated analyses and simulations on utility networks and supplying infrastructures. Included are, amongst others, network hierarchies of arbitrary depth, nesting of network components, and modelling of multi-modal networks. Furthermore, it allows for representing the network components as 3D topographic city objects.

3. Recommendations for standardisation activities

As seen in chapter 2, the standardisation bodies develop and promote not only a single standard but provide a list of standards. Not all of these standards are relevant for STREAMER. Therefore, this deliverable focusses on the following standards:

- buildingSMART – IFC; mvdXML and bSDD (IFD)
- Open Geospatial Consortium – CityGML

3.1 IFC

The STREAMER Workflow is significantly based on open data standards for BIM (IFC) (see Figure 3).

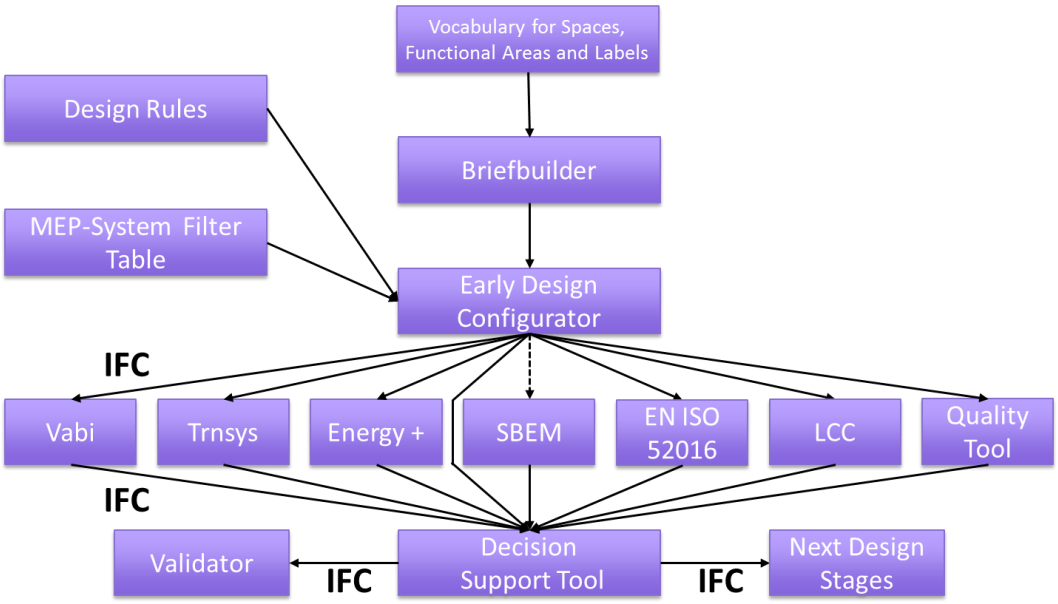

Figure 3: STREAMER workflow in the early design phase

There are three major exchange scenarios in the workflow depicted in Figure 3:

- Exchange between Early Design Configurator (EDC) and Energy Calculation Tools
- Exchange between Energy Calculation Tools and Decision Support Tool
- Exchange between Decision Support Tool and Architectural Design Tools

All of the three scenarios have their specific needs and requirements for exchanging data. While testing this workflow with available, automatically generated and manually modified IFC data, the following deficiencies and shortcomings were detected.

3.1.1 *IfcCoordinateReferenceSystem*

Motivation

The STREAMER project is focusing on healthcare districts and its neighbourhood. In order to integrate IFC and geo data, the correct location and orientation of all spatial data is required. In IFC 2x3, it is only possible to define the location of the building by latitude and longitude and to specify the north direction by a 2D vector. In IFC 4, it is possible to define any coordinate reference system and to give a map conversion to place the local building coordinate system in the world. As in STREAMER different coordinate reference systems are used (e.g. Amersfoort for the Dutch demonstration site) this new possibility in IFC 4 is used to integrate BIM and Geo data.

Current state

In IFC 2x3, *IfcSite* contains the *RefLatitude*, *RefLongitude* and *RefElevation* to determine the global position in the world. In addition, the entity *IfcGeometricRepresentationContext* allows defining the north direction as a vector within the xy-plane.
In IFC 4, a coordinate reference system can be defined by using *IfcProjectedCRS* the only subtype of *IfcCoordinateReferenceSystem*. The transformation between the local engineering coordinate system and the coordinate references system of the underlying map is performed by the entity *IfcMapConversion*.

Problem

In IFC 2x3, it is only possible to use latitude and longitude. It is well defined and easy to use but is limited in terms of GIS integration. IFC 4 additionally supports a more general approach for coordinate reference systems. Nevertheless, geodetic coordinates systems are not covered by the new approach. Therefore, the support of both concepts is still necessary.
The attributes of the entity *IfcCoordinateReferenceSystem* are not in line with the definitions of the EPSG (Name – Code).

Recommendations

The documentation should clearly distinguish between latitude/longitude and projected coordinates. As both can be used simultaneously in a single IFC model, a priority rule has to be given.
There should be a clear statement in the documentation that *IfcMapConversion* is not applicable for latitude and longitude.
The use of *IfcMapConversion* is different from using *RefLatitude*, *RefLongitude* and *RefElevation* (*IfcSite)* and *TrueNorth* (*IfcGeometricRepresentationContext*). This should be precisely described in the documentation.
Using the EPSG code as the name of the coordinate reference system is not consistent with the nomenclature of EPSG itself and other standards like LandXML. In the next version of IFC, this should be corrected.

3.1.2 *IfcOwnerHistory* (schema)

Motivation

In the STREAMER workflow different software applications are used to generate, modify and update the IFC building model. In order to trace the activities properly, the entity *IfcOwnerHistory* is available and used. This entity contains information about the owner, the software application, the change action, the creation date etc.

Current state

In IFC version 2x3, the *IfcOwnerHistory* is a mandatory attribute of *IfcRoot*. Therefore, all entities derived from *IfcRoot* must have an *IfcOwnerHistory*. As specified in IFC, the *Ifc*OwnerHistory only stores the last modification.

In IFC version 4, the *IfcOwnerHistory* is no longer mandatory, but an optional attribute of *IfcRoot*. Even a rule for make the *IfcOwnerHistory* mandatory for the *IfcProject* has been withdrawn. The specification gives a statement, that the use of *IfcOwnerHistory* should be controlled by view definitions and implementer agreements.

Problem

If there is no model view definition and no corresponding implementer agreement, IFC models without any *IfcOwnerHistory* will validate against the schema. Especially if IFC data are enriched during the workflow (like in the case of STREAMER), the *IfcOwnerHistory* is the only entity to identify who has changed or added entities, relations or properties. In IFC 2x3, missing *IfcOwnerHistory* causes a schema error, which can be easily checked. In IFC 4, the *IfcOwnerHistory* is no longer mandatory (even in the standardised view definitions it seems not to be mandatory) and therefore no schema error will occur if *IfcOwnerHistory* is missing. This means, if the *IfcOwnerHistory* is required for a certain process, the existence of *IfcOwnerHistory* must be checked separately.

Recommendations

Theoretically, by defining an own model view definition the problem can be solved. However, in the STREAMER workflow the *IfcOwnerHistory* is a basic concept, which should be made mandatory again.

3.1.3 *IfcOwnerHistory* (usage)

Motivation

The usage of the entity *IfcOwnerHistory* within the STREAMER project raised the question of how to apply the owner history on different elements, like entities, relations or property sets (single properties do not have an *IfcOwnerHistory*). One example for this is the following: If a single property in a property set of an entity is changed, does this influence the owner history of the property set, the owner history of the relation, and/or the owner history of the entity itself? In a real case, where an application changes the property *LoadBearing* of the property set *Pset_WallCommon* of the entity *IfcWallStandardCase* the following possibilities are available:

- As the property itself has no *IfcOwnerHistory,* no owner history will be changed;
- As the property is member of the property set *Pset_WallC*ommon, the *IfcOwnerHistory* of the property set *Pset_WallC*ommon will be changed;
- As the property set is linked to the *IfcWallStandardCase* by the relation *IfcRelDefinesByProperties*, the *IfcOwnerHistory* of *IfcRelDefinesByProperties* will also be changed;
- As the changed property is “modifying” the *IfcWallStandardCase*, the *IfcOwnerHistory* of *IfcWallStandardCase* will also be changed.

Current state

The strategy in using the owner history is depending on the sending application. Some applications maintain the owner history, and other applications ignore existing owner histories and replace them by a new the owner history while exporting the model.
Currently, in the STREAMER project modified and newly added property set including their relations get a new or modified owner history. The owner history of the actual IFC entity will remain unchanged.

Problem

As the property itself has no owner history, only the owner history of the property set can be changed. Therefore, it is possible to trace that a property set has been changed, but not which property has been changed.
In most of the applications, properties cannot be selected and visualised in the graphic. This means, if you want to identify changed entities by a coloured representation of the model, the application itself has to find the entities, which have property sets or relations with a specific owner history.

Recommendations

Currently, the owner history is not really taken serious in the data exchange process. However, as soon as different applications like the STEAMER design and simulation tools add more and more information to the model (enriching the model), the more important the owner history will be.
It is recommended to think about the use of the owner history and give clear instructions in the documentation on how to create the owner history, on which entities are influenced by certain changes and on how to interpret the owner history in the receiving system.

3.1.4 *IfcTimeSeries*

Motivation

Building energy simulation, especially in the design phase, requires weather data not specific for a particular year, but long-term averages for a typical year. Usually, the data set of such a typical meteorological year (TMY) contains 8760 (365 days x 24 hours) values for each property (e.g. temperature, wind speed and horizontal radiation). If using TMY weather data for the simulation, the simulation software usually generates results for each time step (8760). In order to store the simulation results as part of the building information model, the IFC model supports time-stamped data entries by the entity *IfcTimeSeries*.

Current state

IFC supports two kinds of time series: *IfcIrregularTimeSeries* and *IfcRegularTimeSeries*. Both time series require a *StartTime* and an *EndTime*, which request a specific date (year, month, day) and time (hour, minute, second). Depending on a certain year (leap year), the number of days can differ.

Problem

Especially in building simulation, weather data of a typical meteorological year are used. These data sets contain 8760 values for each property and are not related to a specific year. Accordingly, the simulation tools produce the results in the same time steps. Currently the results can only be stored in a time series with a specific start- and end time. In the case of leap year, this can cause problems while importing such time series.

Recommendations

A very simple work around is to select a year in future, which is not a leap year, and use this year as a typical meteorological year.

In the long-term perspective, the IFC Model Support Group (MSG) should discuss concepts to support time series, which are able to cover typical meteorological years and the corresponding simulation results.

3.1.5 *IfcExternalReference*

Motivation

An external reference is the identification of information that is not explicitly stored in the current model or in the project database. Currently, such information may be contained in classifications, documents or libraries [IFC4Add2-2]. The support of web services, as subtype of external references seems not to be intended. In order to support advanced PLM (STREAMER Task 5.2), the direct access to web resources like product catalogues from an IFC model becomes more and more important.

Current state

IfcExternalReference is an abstract super type of [IFC4Add2-2]:

- *IfcClassificationReference* → classification system or source for classification keys or notation
- *IfcDocumentReference* → document
- *IfcExternallyDefinedHatchStyle* → hatching styles
- *IfcExternallyDefinedSurfaceStyle* → surface styles, material libraries for rendering information
- *IfcExternallyDefinedTextFont* → text fonts
- *IfcLibraryReference* → libraries

Problem

Semantically, web services can hardly be classified as classification, document or library. A web service is a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web [WebService2016]. In order to cover the requirements for web services a new IFC entity is needed.

Recommendations

In order to consider the requirements for web services, a new IFC entity named *IfcWebServiceReference* is proposed. As there are different methods to access data, two modelling approaches are possible:

- The *IfcWebServiceReference* will be an abstract super type of more specific services (e.g. *IfcWebServiceGetReference* or *IfcWebServicePostReference*).
- The *IfcWebServiceReference* itself contains parameters, which are required by specific web services.

3.1.6 Attributes "*PredefinedType*" and "*LongName*" of *IfcBuildingSystem* and *IfcDistributionSystem*

Motivation

There is an inconsistency in using the attributes "*PredefinedType*" and "*LongName*" in the subtypes of *IfcSystem*

Current stage

In IFC 4, *IfcSystem* has four subtypes (in IFC 2x3 there are no subtypes): *IfcBuildingSystem*, *IfcDistributionSystem*, *IfcStructuralAnalysisModel* and *IfcZone*.

The subtypes are extended by the following attributes:

- *IfcBuildingSystem* extends *IfcSystem* by *PredefinedType* (Attribute No. 6) and *LongName* (Attribute No. 7).
- *IfcDistributionSystem* extends *IfcSystem* by *LongName* (Attribute No. 6) and *PredefinedType* (Attribute No. 7).
- *IfcStructuralAnalysisModel* extends *IfcSystem* by *PredefinedType* (Attribute No. 6), *OrientationOf2DPlane* (Attribute No. 7), *LoadedBy* (Attribute No. 8), *HasResults* (Attribute No. 9) and *SharedPlacement* (Attribute No. 10).
- *IfcZone* extends *IfcSystem* by *LongName* (Attribute No. 6)

Problem

Using attributes in a different order for different entities is not a technical problem, as the sequence number is given in the schema. However, a systematic and consistent property order reduces the implementation effort and supports an easier understanding of the schema.

Recommendation

At least *IfcBuildingSystem* and *IfcDistributionSystem* can be harmonised in the next version.

	Attribute No. 6	Attribute No.7	Attribute No.8	Attribute No.9	Attribute No.10	Attribute No. 11
IfcBuildingSystem	*LongName*	*Predefined Type*				
IfcDistribution System	*LongName*	*Predefined Type*				
IfcStructural AnalysisModel	*LongName*	*Predefined Type*	*Orientation Of2Dplane*	*Loaded By*	*Has Results*	*Shared Placement*
IfcZone	*LongName*					

The recommendation was reported by KIT on 16th November 2015 and fixed (*IfcBuildingSystem*, *IfcDistributionSystem*) by Thomas Liebich, AEC3, on 12th May 2016, for the future version IFC 5 alpha.

3.1.7 Usage of *IfcSpatialZone*

Motivation

"A spatial zone is a non-hierarchical and potentially overlapping decomposition of the project under some functional consideration. A spatial zone might be used to represent a thermal zone, a construction zone, a lighting zone, a usable area zone. A spatial zone might have its independent placement and shape representation [ISO16739], [IFC4Add2]."

In STREAMER, zones are used to represent "Functional Areas". In order represent "Functional Areas" with its own geometric representation, it is intended to use *IfcSpatialZone* instead of *IfcZone*.

Current state

In the previous version of IFC (IFC 4 Add1 valid until 2016), it was not possible to use *IfcSpatialZone* in an IFC model without violating the IFC schema. None of the possible relations to connect an *IfcSpatialZone* with other spatial elements could be used to create a correct IFC model.

Problem

Using *IfcSpatialZone* means to use a spatial element subtype, which is outside the typical hierarchical relationship (*IfcSpatialStructureElement*) project -> site -> building -> story -> space. Therefore, it is important to agree on the relationships between *IfcSpatialZone* and the other spatial and physical elements. The alternatives are:

- Using *IfcSpatialZone* as an *IfcSpatialElement* by using *IfcRelAggregates* (for connecting to a storey and to connect spaces) will not validate ('*IFCSPATIALSTRUCTUREELEMENT.WR41*').
- Using *IfcSpatialZone* as a *IfcGroup* by using *IfcRelServicesBuildings* (for *IfcSpatialZone* to storey) and *IfcRelAssignsToGroup* (for spaces to *IfcSpatialZone*) will not validate (incompatible assignment: *'RelatingGroup'*)
- Using *IfcSpatialZone* as an *IfcSpatialElement* by using *IfcRelReferencedInSpatialStructure* (for connecting to a storey and to connect spaces) will not validate ('*IFCRELREFERENCEDINSPATIALSTRUCTURE.WR31*').

Recommendations

It is recommended to use *IfcRelReferencedInSpatialStructure* between physical elements and *IfcSpatialZone*. It is also recommended to use this relationship to connect the *IfcSpatialZone* to spatial structure elements (site, building, building storey, space).

In order to allow a valid IFC instance model, the *IfcRelReferencedInSpatialStructure* WHERE rule (AllowedRelatedElements) has to be relaxed to allow *IfcSpace* to be a related element (allow spaces to be related to an *IfcSpatialZone*). Add documentation to describe the use of *IfcRelReferencedInSpatialStructure* to connect elements and spaces to the spatial zone, and to connect the spatial zone to spatial structure element (building or storey).

The recommendation was reported by KIT on 28th March 2015 and fixed by Thomas Liebich AEC3 on 28th June 2016 for IFC 4 Add2.

3.1.8 Space Boundaries for *IfcExternalSpatialElement*

Motivation

External spatial elements define regions at the building site. Examples for *IfcExternalSpatialElement* are the air space around the building (usually without an own shape representation) or the ground area, which is a physical space with a corresponding shape representation.
IfcExternalSpatialElement can have space boundaries, which represent the outer shell of the building. In the STREAMER project these space boundaries are of high interest, as they represent the correct outer surfaces for energy calculation, and by using the predefined types of *IfcExternalSpatialElement* they represent the boundary conditions (against air, earth or water), as well.

Current state

IfcExternalSpatialElement can be used as an external space. There is no further documentation for using the space boundaries in a different way than the space boundaries for *IfcSpace*.

Problem

Even if it is not clearly stated in the documentation, the space boundaries of *IfcSpace* are exported as surfaces with normal directions pointing outside of the space. Applying this to the *IfcExternalSpatialElement* means that the normal directions of the boundary surfaces are pointing inside of the building. This means, if using this surface for energy calculation the surface normal have to be reversed.

Recommendations

It would be good to clearly describe the normal directions of boundary surfaces in both case, for *IfcSpace* and *IfcExternalSpatialElement,* in the specification. If treated in the same way (normal directions pointing outside of spaces), it should be mentioned, that for the specific applications the surface normal direction has to be reversed.

3.1.9 Property set "*Energy Consumption*" for *IfcBuilding* and / or *IfcSite*

Motivation

Especially, when considering healthcare districts (usually more than one building), it is desirable to add energy consumption values (in KWh) of corresponding utility resources, like gas, water or electricity, to the buildings and as aggregated values to the site.

Current state

IFC 4 allows adding a property set *Pset_UtilityConsumptionPHistory* to the building. This property set contains properties for the consumption of *Heat*, *Electricity*, *Water*, *Fuel* and *Steam*. By definition, this property set can only be used by *IfcBuilding*.

Problem

If considering districts, like STREAMER's healthcare districts, it is not possible to store the overall consumption values on the site level. It is only possible to store the consumption values on building level and the receiving application has to add up the single building consumption values.

Recommendations

It is recommended to extend the use of the property set *Pset_UtilityConsumptionPHistory* for *IfcSite* as well. This allows storing consumption values for a complete site, if there are more than one building.
If a further differentiation of the consumption values is of interest in general, it is recommended to look for a concept allowing multiple consumption values, which then must be classified.

3.1.10 Property set "*Energy Demand*" for *IfcBuilding* and / or *IfcSite*

Motivation

For dimensioning HVAC facilities in the building and the supply of the complete district, it can be useful to store the demand (in KW) of utility resources, like gas, water or electricity, on building and site level.

Current state

In IFC 2x3 and IFC 4, there is no property set known to store the demand of utility resources.

Problem

Knowing the energy consumption will not be enough for dimensioning the HVAC equipment in the building and on the district level. In the case, it is more useful to have maximum demand values within a year, a month or a week.

Recommendations

It is recommend discussing the need of demand values of buildings and district with corresponding experts. If agreed, it is proposed to add a new property set, accordingly to the *Pset_UtilityConsumptionPHistory.*

3.1.11 Property set "*Indoor Air Quality*" for *IfcSpace*, *IfcZone* and *IfcSpatialZone*

Motivation

The indoor air quality is an important parameter for hospital rooms. Especially in the "hot floor" area, special regulations have to be considered.
In the STREAMER "Label" concept, especially the Label "HygienicClass" has a strong relation to supply air quality, air tightness and airflow. As the application of STREAMER labels is specific for health care district, a general place for storing indoor air quality data is missing.

Current state

In the current IFC version, there is no general property set for indoor air quality. Only for airside HVAC systems, the property set *Pset_AirSideSystemInformation* allows two parameters for infiltration.

Problem

If there are special requirement for indoor air quality, IFC offers no property sets to store relevant data. In the STREAMER project, the use of labels implies a certain level of indoor air quality. However, commercial application, which might require indoor air quality data (e.g. energy calculation tools), cannot use STREAMER labels but need corresponding properties.

Recommendations

It is recommend, to define a property set (or if necessary property sets) for parameters relevant for describing the indoor air quality. As basis for the property definition, the following standards can be considered:

- EN 13779:2007: Ventilation for non-residential buildings
- DIN EN 779: Particulate air filters for general ventilation
- EN 1822-1: High efficiency air filters (EPA, HEPA and ULPA)

3.1.12 Property set for Life Cycle Assessment (LCA)

Motivation

With IFC 4, new property sets for environmental impact factors are defined. In parallel, the European community was launching “The Life Cycle Data Network (LCDN)”. The network is providing an infrastructure for the publication of LCA datasets. The LCA datasets are defined in the ILCD format [ILCD2014]. In order to use the LCA datasets from the LCDN effectively, it would be preferable to harmonise the ILCD data sets and the IFC property sets.
As the Life Cycle Assessment is an important part of the overall performance of the building, it is also an import aspect in the STREAMER project.

Current state

IFC 4 offers two property sets for environmental impacts: *Pset_EnvironmentalImpactIndicators* and *Pset_EnvironmentalImpactValues*. ILCD offers data sets, which are suitable for performing a Life Cycle Assessment (LCA). Some of the properties are equivalent in both declarations (e.g. IFC: ClimateChangePerUnit – ILCD: Climate change), but there are properties missing in IFC.

Problem

Experiments concluded that, a one to one mapping of the available properties in IFC and ILCD is not trivial. In order to get a final evaluation, a deeper look into the ILCD and IFC standards is necessary. Especially, it has to be reviewed if the properties available in IFC can support the process of a life cycle assessment.

Recommendations

The modelling group of IFC should have a look to the activities of the LCDN. In order to harmonise the LCA properties for performing a life cycle assessment, it would be advisable to consult LCA experts. Environmental Product Declaration (EPD) for BIM (construction products and services) is a New Work Item Proposal (NWIP) led by ISO TC59 SC17 WG3. It proposes to adopt and adapt BS8541 part 6 [BS 8541-6]. The relationship between ILCD data (which lacks explicit base quantities) and its format (which is not transportable) and product data in IFC format will be explored.

3.1.13 Property set for "Program of Requirements"

Motivation

In many cases, in the early design phase a "Program of Requirements" (PoR) for spaces is generated. In the STREAMER project, this is the basis of all following planning activities. Therefore, it is desirable to store the information of the PoR permanently in the IFC model, in order to preserve it for the complete product life cycle. The corresponding properties must be defined.

Current state

On space level, there are several property sets, which can be used for space requirements:

- ***Pset_SpaceCommon*** → *GrossPlannedArea*, *NetPlannedArea*
- ***Pset_SpaceCoveringRequirements*** → different requirement for space covering
- ***Pset_SpaceFireSafetyRequirements*** → different requirement for fire safety
- ***Pset_SpaceLightingRequirements*** → *ArtificialLighting*, *Illuminance*
- ***Pset_SpaceOccupancyRequirements*** → different requirement for work activities occurring or expected to occur within one or a set of similar spatial structure elements, e. g. *IsOutlookDesirable*
- ***Pset_SpaceThermalRequirements*** → temperature, AC, ventilation rate, humidity etc.

Problem

In order to cover the specific case of a STREAMER program of requirements, several IFC entities and property sets must be used. The RoomType can be mapped into *IfcSpaceType*, the FunctionalAreaType can be used to create an *IfcZone* and for the property Area the *NetPlannedArea* of the *Pset_SpaceCommon* is available.

Unfortunately, according to the specification, the property sets representing the requirements can be used for the requirements coming from a space program in early project phases and can be used to define the room book information in later phases (Definition buildingSMART: "Properties common to the definition of covering requirements of *IfcSpace*. Those properties define the requirements coming from a space program in early project phases and can later be used to define the room book information, if such coverings are not modelled explicitly as covering elements") [IFC4Add2-1]. Especially, if there are deviations between the required settings and the actual implementation, the requirements are usually overridden.

Recommendations

In order to avoid overriding properties in different design phases, the property sets can be separated for each design phase or they are allowed to occur multiple times for different life cycle phases. Solutions should be discussed in the buildingSMART IFC Modelling Group.

If a versioning system is available or a database will store different versions of the building model this problem can be solved by creating different versions.

3.2 mvdXML

The mvdXML format enables to specify a Model View Definition (MVD), which is a subset of IFC that is required to fulfil the needs of one or more use cases. mvdXML is an open standard published by buildingSMART and was available as release 1.0 in the beginning of the STREAMER project. Release 1.1 was in draft stage. MVDs are used in this project to specify and check the data exchange in the STREAMER workflow (see also deliverable D5.2).

The type of model checks that are in focus of STREAMER enable to control if all requested data is contained in an IFC BIM file or if something is missing due to incomplete data inputs or wrong data exports. It can be done both on sender and receiver side using for instance the freely available XBIM tool, which has been extended by mvdXML checking capabilities with support of the STREAMER project.

This chapter is discussing three topics related to MVD developments:

1. Improvement of the MVD specification
2. Proposed changes that have been considered in the final release of mvdXML 1.1 already
3. Proposed changes for a next mvdXML release to improve model checking capabilities

3.2.1 MVD for energy related properties

Motivation

MVD development should support three types of defining data requirements, namely by attributes, grouping and classification.

Current state

IFC has the ability to hold attribute information in three ways, but only properties seem to be supported by existing tools.

- An attribute can be named, described and assigned a value. Usually the attribute will be held in a property set, which is associated to an object.
- A classification code may be associated to the object. The name of the classification table or system can also be included.
- An object can be assigned to a named group (often a system or zone) which implies a value for an attribute, or the attribute may be assigned to the group. The groups 'object type' may indicate the topic, and other groups with the same 'object type' may include objects with different values for a similar attribute.

Problem

Alternative representations like classification or grouping can lead to reduced file sizes and better data management than property set based representation of labels. However, supporting alternative representation will not only increase implementation efforts for receiving applications but also will increase the management effort of requirement specifications. Additionally, there is a risk of inconsistencies if conflicting alternative representations are given.

	Attribute	Classification	Group
Topic:	Name	Table Name	Object Type
Semantics:	Description	Description	Description
Value:	Nominal Value	Code	Name

Recommendations

The STREAMER partners decided to go with properties for the representation of PoR labels. There are mainly practical reasons. Further clarification is needed which kind of properties shall be used. There are two options:

- Store labels as *IfcPropertySingleValue*, which is a more flexible solution supported by most available tools. The drawback of this solution is that use label values need to be checked by the mvdXML specification.
- As alternative, it is recommended to switch to *IfcPropertyEnumeratedValue*, which enables to encode all allowed options within the IFC file and thus enables more generic user interfaces. Additionally, proper use of enumeration values does not have to be checked by mvdXML.

3.2.2 mvdXML 1.1 – Support logical tree of rules

Motivation

mvdXML 1.0 is encoding checking rules in a single string parameter. In mvdXML 1.1 a formal grammar has been introduced that adds features like OR, XOR, NOR combination and grouping of logical statements. While this grammar is technically very powerful, it easily becomes complex and difficult to read in many practical examples like for instance the check of properties. The motivation of this proposal is to increase readability of logical expressions by reducing the length of string-encoded statements and to avoid nesting of expression using parenthesis.

State of mvdXML 1.0

A complex string encoded rule is shown below for the check of the property *Pset_WallCommon.Status*. The rule checks if the property is attached either to the occurrence object or a potentially given type object.

```
<TemplateRule Parameters="(Set[Value]='Pset_WallCommon' AND
PropertyName[Value]='Status' AND Value[Exists]=TRUE) OR ((Set[Value]!='Pset_WallCommon'
AND PropertyName[Value]!='Status') AND (T_Set[Value]='Pset_WallCommon' AND
T_PropertyName[Value]='Status' AND T_Value[Exists]=TRUE))"/>
```

Problem

Encoding of rules can be automated to some extent, for instance by using a tool like BIM*Q hiding complexity from users by providing a simple user interface for the check of properties. However, in many cases a direct use of the syntax is expected so that a more user friendly, structured solution is necessary.

Recommendations

A solution was proposed to use a logical tree of string-encoded rules as shown in the example below. It defines the check for the same property *Pset_WallCommon.Status*, but is more user friendly than the original solution, which however is still possible.

```
<TemplateRules operator="or">
  <TemplateRule Parameters="Set[Value]='Pset_WallCommon' AND
                              PropertyName[Value]='Status' AND Value[Exists]=TRUE"/>
  <TemplateRules operator="and">
        <TemplateRules operator="nor">
          <TemplateRule Parameters="Set[Value]='Pset_WallCommon' AND
                                    PropertyName[Value]='Status'"/>
        </TemplateRules>
        <TemplateRules operator="or">
        <TemplateRule Parameters="T_Set[Value]='Pset_WallCommon' AND
                           T_PropertyName[Value]='Status' AND T_Value[Exists]=TRUE"/>
        </TemplateRules>
  </TemplateRules>
</TemplateRules>
```

3.2.3 mvdXML 1.1 – Constraints for applicable entities

Motivation

Each checking rule must include a criterion that enables to differentiate between applicable and non-applicable objects.

State of mvdXML 1.0

The select criteria in mvdXML 1.0 is the entity type, which for instance would allow defining a rule for all objects being an instance of the *IfcWall* entity.

Problem

In many cases it is necessary to further refine this select criteria, for instance to select load bearing walls only. In IFC, it would for instance mean to select all instance of *IfcWall* where the property *Pset_WallCommon.LoadBearing* is *true*. Such level of select statements are not possible if only applicable entities can be defined. In STREAMER, such refinement is necessary to differentiate room types based on given properties. A corridor is for instance not considered in the PoR, but later generated by the EDC. Checking of all rooms would therefore fail for generated corridors.

Recommendations

It is recommended to add an optional *Applicability* section to *ConceptRoot* that enables to refine the entity-type based selection of instances. This enables to exclude Corridors from checking PoR requirements. The example below shows the selection of load bearing walls as described in the problem statement.

```
<ConceptRoot uuid="00000003-0095-0000-0000-000000000917" name="Load bearing wall"
applicableRootEntity="IfcWall">
  <Applicability>
        <Template ref="00000000-0000-0000-0001-000000000001"/>
        <TemplateRules operator="and">
          <TemplateRules operator="or">
                <TemplateRule Parameters="Set[Value]='Pset_WallCommon' AND
                                Property[Value]='LoadBearing' AND Value[Value]=FALSE"/>
                <TemplateRules operator="and">
                  <TemplateRules operator="nor">
                        <TemplateRule Parameters="Set[Value]='Pset_WallCommon' AND
                                                  Property[Value]='LoadBearing'"/>
                  </TemplateRules>
                  <TemplateRules operator="or">
                        <TemplateRule Parameters="T_Set[Value]='Pset_WallCommon' AND
                              T_Property[Value]='LoadBearing' AND T_Value[Value]=FALSE"/>
                  </TemplateRules>
                </TemplateRules>
          </TemplateRules>
        </TemplateRules>
</Applicability>
```

3.2.4 mvdXML 1.1 – Modularisation of concept template definitions

Motivation

An important feature of mvdXML are configurable concept templates that can hide complexity when defining checking rules. A concept template for instance defines the path that needs to be traversed to extract a piece of information attached to a building object like a wall, column etc. Such piece of information can be tagged by a parameter, which is later used to quickly specify a checking rule.
The setup of concept templates itself shall enable to reuse other concept templates to avoid recurring definitions. This would improve the management of configurable concept templates.

State of mvdXML 1.0

Reuse of concept templates was not supported.

Problem

Reuse of concept templates would be for instance relevant for checking properties as described in chapter 3.2.1. The IFC mechanism to attach a property to an occurrence or a type object is exactly the same. Accordingly, it would be useful if that functionality can be specified once in a concept template and then reused for occurrence and type objects. Besides proper referencing of such partial concept templates, it is necessary to guarantee uniqueness of configurable parameters.

Recommendations

The proposed and accepted change for mvdXML 1.1 was to introduce partial concept templates and the use of an optional prefix for configurable parameters. The main concept template for checking properties is shown below. An *EntityRule* now enables to define a reference to other concept templates with an optional *IdPrefix* attribute that is added to all *RuleID* parameters of the referenced concept template.
The partial template referenced by *IfcRelDefinesByProperties* is also used the by partial template *IfcRelDefinesByType* so that the prefix *"T_"* was added to guarantee uniqueness of the parameters `Set`, *Property* and *Value*.

```
<ConceptTemplate uuid="00000000-0000-0000-0001-000000000001"
name="ProductConceptTemplate" applicableSchema="IFC4" applicableEntity="IfcProduct">
  <Rules>
       <AttributeRule AttributeName="IsDefinedBy">
         <EntityRules>
              <EntityRule EntityName="IfcRelDefinesByProperties">
                <References>
                     <Template ref="10000000-0000-0000-0001-000000000001"/>
                </References>
              </EntityRule>
              <EntityRule EntityName="IfcRelDefinesByType">
                <References IdPrefix="T_">
                     <Template ref="10000000-0000-0000-0001-000000000002"/>
                </References>
              </EntityRule>
         </EntityRules>
</AttributeRule>
```

3.2.5 mvdXML 2 - Consider units for value checks

Motivation

Checking of data requirements is currently focused on checking the existence of data (attributes, properties, quantities, relationships between objects). In the STREAMER project it is also important to check the correctness of values, which means to validate if given labels are according to allowed enumeration values. If measured values like for instance the area of a room must be validated, the used unit of measurement becomes important and must be considered.

Current state

mvdXML is a generic specification that is independent from any model semantics and therefore does not include the functionality to specify the unit to be used for model checking. The most practical solution at the moment is to request for a global unit setting of a measured value type. The check of measured values is then defined against this unit. Accordingly, two independent checks need to be defined: (1) check the global unit, e.g. area values must be given in square meters and (2) check the value, e.g. the real value of the net room area must be greater than 10.0;

Problem

If values are given in a different unit, e.g. metre instead of milli-metre, the check of the measured value fails even if the it is semantically correct (x > 1000 milli-metre will fail if x = 1.1 metre).

Recommendations

STREAMER does not have a preferred way to solve that issue, as it needs further discussion how to best integrate into mvdXML. Main question is whether mvdXML should be kept generic or if unit settings should be added to the mvdXML specification. Ideally, the check of measured values can be defined in one selected unit, and does not have to consider all possible units.

3.2.6 mvdXML 2 - Consider tolerance for check of real values

Motivation

The check of real values should enable to set a tolerance to avoid wrong checking results caused by rounding errors. If a value must be equal to 1.0, then any small deviation like 0.99999999 is reported as a fail of the check.

Current state

Value checks where rounding errors might be an issue must define a range of allowed values. A statement like x=1.0 must be encoded to check a range of values, like for instance x>0.95 AND x<1.05 considering a tolerance of +/-0.05.

Problem

Specification and management of value checks becomes more complicated and less readable. It is not always obvious that a specific value is checked with an allowed tolerance.

Recommendations

It is suggested to set a tolerance value for the check of real values. The following requirements should be supported:

- Tolerance value should be optional
- Tolerance value should allow to differentiate between plus and minus tolerance
  (in some cases they need to be different)
- Tolerance value should be definable in absolute and percentage value
- Tolerance value may be given in a global setting or refined for a specific check.

3.2.7 mvdXML 2 - Global existence of instances

Motivation

Model checking is based on two parts (1) a select criteria that differentiates between applicable and non-applicable objects and (2) a checking criteria that must be fulfilled by applicable objects. Global existence of objects like the STREAMER rule "*there must be a space object*" is currently not checkable.

Current state

mvdXML 1.1 does not support a global existence check of objects. Such kind of checks are only possible via references that must be attached to other objects like for instance *IfcProject*, *IfcBuilding* or *IfcBuildingStorey* (local existence of references + type check of referenced objects).

Problem

Above described work-around depends on existence of other objects (in case of *IfcProject* there is a global rule that ensures that, otherwise the IFC is not compliant with the schema) and availability of references used for checking the existence of other objects.

Recommendations

Global existence of objects should be added as a checking feature to mvdXML. A straightforward extension would be to check size of applicable entities. For instance a default parameter *"Self"* could be introduced to check either (A) the number of applicable entities or (B) the type of applicable entities. Both examples are shown below.

```
<ConceptRoot uuid="00000018-0077-0000-0000-000000005155"
             name="0 Room and Room type" applicableRootEntity="IfcSpace">
  <Concepts>
       <Concept uuid="00000018-0077-0000-0000-000000005157" name="... ">
              <Template ref="e26040e8-82e2-4f6a-bc63-ac8e6da2d0ae"/>
              <TemplateRules operator="and">
                     <TemplateRule Parameters="SELF[Size]>0"/>
              </TemplateRules>
       </Concept>
  </Concepts>
</ConceptRoot>

<ConceptRoot uuid="00000018-0077-0000-0000-000000005155"
             name="0 Room and Room type" applicableRootEntity="IfcRoot">
  <Concepts>
       <Concept uuid="00000018-0077-0000-0000-000000005157" name="... ">
              <Template ref="e26040e8-82e2-4f6a-bc63-ac8e6da2d0ae"/>
              <TemplateRules operator="and">
                     <TemplateRule Parameters="SELF[Type]='IfcSpace'"/>
              </TemplateRules>
       </Concept>
  </Concepts>
</ConceptRoot>
```

3.2.8 mvdXML 2 - Check of set values – differentiation between existence and for all

Motivation

The check of *SET* values is basically testing if one of the set values fulfil the requirement. This enables to easily check existence of properties, but does not support to check the consistency of all set values.

Current state

There some agreements for *SET* (including *LIST*, *BAG*, *and ARRAY*) values that need further clarification and revision.

Problem

It seems that checking consistency of all SET values needs a negative statement that excludes wrong values from the data set. If all values of a set must be greater than 1.0, then values that are lower or equal to 1.0 must be excluded (see examples below).

```
<ConceptRoot uuid="00000018-0077-0000-0000-000000005155" name="…"
             applicableRootEntity="IfcSpace">
  <Concepts>
       <Concept uuid="00000018-0077-0000-0000-000000005157" name="…">
              <Template ref="e26040e8-82e2-4f6a-bc63-ac8e6da2d0ae"/>
              <Requirements>
                     <Requirement applicability="import"
exchangeRequirement="00000018-0077-0078-0000-000000000000" requirement="excluded"/>
              </Requirements>
              <TemplateRules operator="and">
                     <TemplateRule Parameters="SetOfLength[Value]<=1.0"/>
              </TemplateRules>
       </Concept>
  </Concepts>
</ConceptRoot>

<ConceptRoot uuid="00000018-0077-0000-0000-000000005155" name="…"
             applicableRootEntity="IfcSpace">
  <Concepts>
       <Concept uuid="00000018-0077-0000-0000-000000005157" name="…">
              <Template ref="e26040e8-82e2-4f6a-bc63-ac8e6da2d0ae"/>
              <Requirements>
                     <Requirement applicability="import"
exchangeRequirement="00000018-0077-0078-0000-000000000000" requirement="mandatory"/>
              </Requirements>
              <TemplateRules operator="nor">
                     <TemplateRule Parameters="SetOfLength[Value]<=1.0"/>
              </TemplateRules>
       </Concept>
  </Concepts>
</ConceptRoot>
```

Recommendations

Improve documentation of existing functionality by providing further examples and enable/simplify the differentiation between existence and for all checks.

3.2.9 mvdXML 2 - Further grouping of requirements

Motivation

In many STREAMER cases, there are alternatives for providing the requested data. For instance, properties can be attached to occurrence or type objects leading to complex logical (X)OR expressions (see also improvements of mvdXML 1.1 described in chapter 3.2.1). Another example for the need of alternative representations is given in chapter 3.2.1.

Current state

The rule for checking a particular information must be encoded in mvdXML in one `Concept` element that is defined for a set of applicable entities. It means that all alternatives must be encoded in this single rule leading to a logical tree of expressions.

Problem

The logical tree of expressions as introduced in mvdXML 1.1 is already leading to improved readability and maintenance of rules. However, it could be further simplified if rules can be reused similar to partial concept templates.

Recommendations

Further modularisation of rules is recommended.

3.3 bSDD (IFD)

"The buildingSMART Data Dictionary (bSDD) is a library of objects and their attributes. It is used to identify objects in the built environment and their specific properties regardless of language, so that "door" means the same thing in Iceland as it does in India.
The Data Dictionary is open (freely accessible, according to ISO 12006 part 3 [ISO12006-3]) and international (accommodates any ISO 639 [ISO639] recognised language or dialect), allowing architects, engineers, consultants, owners and operators on one side and product manufacturers and suppliers on the other from all around the world to share and exchange product information. When everyone shares the same language, the building process becomes more efficient.
The bSDD is not for specific instances of objects, but for general terms. Software developers can build on top of these definitions to create specific objects [bSIDD2016].

3.3.1 General bsDD recommendation

Motivation

Collaboration and sharing of information is hindered if terms used are not consistent. Within any schema and data format, the ability to match objects may depend on matching semantic vocabulary terms.

Current state

Most applications and domains have implemented distinct sets of terms and, although it has been long anticipated, it is only recently that the need to consistency has emerged at an operational level.

Problem

Any solution should support convergence of terms, but must also allow for the existing state with the continued variations of language, of disciplines and of proprietary applications

Recommendations

bSDD should host attribute synonyms, including proprietary attributes (Revit, SBEM etc.). There is a historic backlog of attribute names that have been used in proprietary applications and existing ISO and CEN standards. At the same time as making clear recommendations for preferred terms, including the IFC property set recommendations, the bsDD should hold these historic terms so as to support interoperability and a transitional process.
bSDD should host classification codes as synonyms for specific objects. Whilst classification tables are presented as groups of types of objects, the bsDD should hold the distinct codes found in different tables as synonyms for the object types. This would support the mapping of objects found in BIM models to multiple classification tables.

3.4 CityGML

The planning of newly designed or retrofitted hospital buildings requires detailed information on the building's neighbourhood (see Figure 4). This includes existing hospital buildings and the existing infrastructure (traffic, energy, and telecommunication) on the hospital site and in the surrounding city. In order to be efficiently used in design processes, corresponding data must be available in standardised formats.

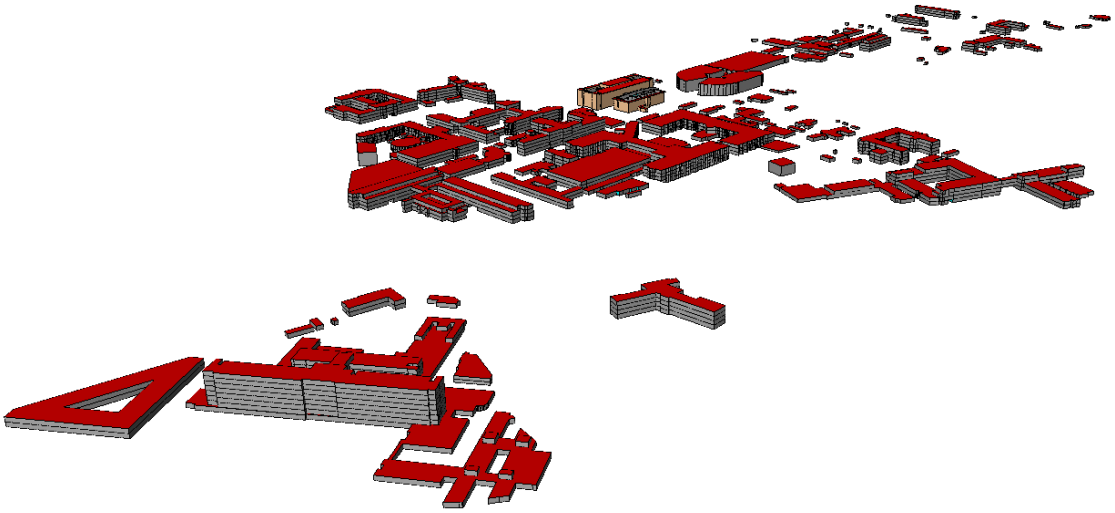

Figure 4: CityGML LoD 2 model of Azienda Ospedaliero-Universitaria Careggi, Firenze, Italy

In the area of 3D GIS data on site or city level, CityGML is the most frequently used standard. However, during the STREAMER project a lot of deficiencies and shortcomings in the available CityGML-based data became obvious. Central information needed for simulations or assessments are either not represented in the CityGML datasets, or the needed information cannot be interpreted in an unambitious manner. In many cases, this is caused by shortcomings and lacks in the CityGML standard. This section of the deliverable therefore summarises a number of recommendations for improving the standard.

3.4.1 CityGML Documentation

Motivation

A data exchange format can only support the interoperability of software applications when all used concepts are clearly defined. For an object oriented data model like CityGML this especially means, that the semantic meaning of all classes, attributes and relations must be specified. In some cases, e.g. with attributes defined by code lists, also the semantic meaning of attribute values must be specified. In order to be efficiently used, this documentation must be clearly structured and easily accessible.

Current state

The documentation of CityGML version 2.0 consists of one monolithic pdf document with 328 pages. The document combines information on three different levels: (1) Informal descriptions of basic CityGML concepts (e.g. the Level of Detail (LoD) concept); (2) normative descriptions of the different modules (e.g. *Core Module*, *Building* …) and the corresponding CityGML classes, attributes and relations, and (3) extensive listings of the CityGML schema files.

Problem

The description of the basic concepts is partly vague and ambiguous. For many classes, attributes and relations, the specification is incomplete. The formal structure of the specification document is weak, which makes it very difficult and time consuming for a user or implementer to find a specific definition or explanation.

Recommendations

A general revision of the whole specification is needed, aiming at

- elimination of vagueness and ambiguities in the description of certain concepts (e.g. the LoD concept), and
- adding missing definitions of CityGML classes, attributes and relations.

Furthermore, the monolithic specification document should be separated into three separate documents:

- A short and informal description of the basic concepts,
- A clearly structured and easily accessible Feature Catalogue, providing formal definitions for every CityGML class, attribute, relation, and code list or enumeration item.
- A clearly structured document containing all additional conformance requirements, expressed (as far as possible) in a formal language.

3.4.2 CityGML file extension and archive files

Motivation

In the STREAMER project, different XML-based data formats have been used in parallel. Besides CityGML models, the STREAMER workflow integrated XML-based design rules as well mvdXML and gbXML files. In order to support an easy identification of a specific file format, a unique file extension would be helpful.

CityGML models normally contain references to externally stored information like textures. In order to facilitate the distribution of such models, a file archive containing the complete content of the model is desirable.

Current state

OGC and the CityGML standard do not prescribe the file extensions of CityGML model files. Usually they have the extension *.xml or *.gml. In both cases, the user or the interpreting application cannot directly imply that the content of the file is a CityGML model.

If a CityGML model is using textures, they are not store directly within the CityGML file. Like many other data exchange formats, the textures are stored in separate files (*.jpg, *.png etc.). If these files are stored locally (which mostly is the case), the data exchange must transfer the texture files as well. It is in the responsibility of the sending application to collect and send all files. Usually the sending system will generate an archive and distribute this archive.

Problem

In contrast to other file formats (IFC → *.ifc, *.ifczip; Google Earth → kml, kmz; etc.), CityGML has no predefined file extensions. Especially when working with different GML application schemata, it is not easy to identify CityGML files. As not all data are stored and exchange via database functions, file exchange is still necessary.

Especially in the case of textured models, it is the responsibility of the sending application to supply all data, e.g. in form of an archive. The receiving application then has to unpack the archive, before he can import the model.

Recommendations

It is recommended to define a mandatory file extension for CityGML files. It is also recommended to define an archive format for exchanging CityGML files including additional content like textures, local documents, drawings etc. In order to read such an archive directly from an application, it is recommended to define a file extension for the archive as well.

To minimise the implementation efforts, the CityGML Standard Working Group should select a common archive format, which provides an API and is freely usable (see also ifczip).

3.4.3 Harmonisation of feature names and feature conceptualisation

Motivation

Semantic data models should be developed in such a way, that the naming of features and properties and the inheritances structure is following a consistent, logical and understandable way. Furthermore, the underlying concept, which real world object or which part of a real world object is represented by a feature, must be specified unambiguously and clearly described. This makes not only the modification and extension of model itself more easily and effectively, but also the use of model more clearly and safe.

Current state

Especially, the naming of boundary surfaces (*_BoundarySurface* and derived features) and openings (*_Opening* and the derived features *Door* and *Window*) in the building module has deficiencies. Except of *ClosureSurface*, there are distinct features for the different parts of the building's exterior shell and the interior shell of a room, but this distinction is not reflected in the naming of the *_BoundarySurface* classes. The features Door and Window represent – as all *_BoundarySurface* classes – the exterior boundary of the corresponding physical objects, which again is not reflected in the feature names.

Problem

The naming and conceptualisation of CityGML features is partly inconsistent, which makes implementations in STREAMER unnecessary difficult.

Recommendations

A general revision of the standard concerning the naming of features and properties is recommended.

3.4.4 Rethink of the Modularisation of CityGML

Motivation

One method to structure a data model is to define "resources", which bundle properties to describe specific aspects of real world objects. One example for a resource is the material definition, which can be used for walls, slabs or roofs. If necessary and desired, the CityGML features representing real word objects should be related with "thematic domains". Examples for such domains in the context of cities are buildings, infrastructures and vegetation.

Current state

CityGML currently is subdivided into 16 modules, each of them with its own namespace. The module structure does not reflect the differentiation between features representing real world objects and features (resources) used to describe these objects. In addition, some of the modules just contain a single feature (CityFurniture, LandUse etc.).

Problem

Following the current concept, new resources like material, stakeholders or costs would need to define new modules with new namespaces. The intension of modularisation is to simplify the maintenance and the implementation of the model and to allow different versions of a module identified by a namespace. However, this intention will be missed if the number of modules is not limited.

Recommendations

It is recommended to define resources for the data model, which can be used, to describe certain aspects of real world objects. Potential resources are geometry, material definitions, appearance (presentation) information, topological information, quality information etc. In addition, it is recommended to structure the real world objects in thematic domains and if necessary in subdomains.

3.4.5 Improvement of the CityGML LoD Concept

Motivation

One of the most prominent and most frequently cited concepts of CityGML is the Level of Detail (LoD) concept. It enables that city objects can be represented in up to five variants, differing in geometrical detailing and semantical structuring. This concept is crucial for many applications, especially for simulations. On the one hand, such applications requires a minimal amount of information in order to produce reliable results. On the other hand, a model containing too detailed geometry or a too complex semantical structure may be inappropriate for certain simulation tools,

Current state

In CityGML, most features have multiple geometry properties, whose names start with "LoD X" (X = 0, 1, 2, 3, 4). The "LoD X" representation of an object is defined by all its geometry properties with the corresponding prefix. The supported LoD differ among the CityGML features. Some like e.g. the Building feature support all five levels, while e.g. features representing the building's interior (*Room*, *BuildingFurniture*, *and IntBuildingInstallation*) only have "lod4" properties.

Problem

The current CityGML LoD concept has a number of weaknesses and deficit, which are extensively discussed in [Benner2013]. The most important ones are:

- The weak and ambiguous definition of LoD, especially for features not belonging to the Building module;
- The strict coupling of geometric and semantic complexity;
- A severely restricted model for building's interior components; and
- Missing metadata characterising different LoD.

Recommendations

Adapt the CityGML LoD concept according to the proposals documented in [Benner2013].

3.4.6 Representation of the spatial structure of a city

Motivation

Cities, especially bigger cities or mega cities are usually subdivided into spatial structures. Rural areas, small towns or townships are often combined in association communities. Even for visualisation purposes, it would be helpful to represent such spatial structures (hierarchical or non-hierarchical) in the 3D city model. For other applications, like energy estimation, supply and disposal, or administration, the spatial subdivision of cities is even more important. Typical examples for a spatial subdivision of cities are: Districts within cities/towns, locality divisions, postal divisions of cities or suburbs.

Current state

The root element of a CityGML model is *CityModel*. The *CityModel* is a collection of *_CityObject* and optional *Appearance* objects. Between *CityModel* and *_CityObject,* there are no other structuring elements. The only possibility to group objects is the use of the general CityGML feature *CityObjectGroup*. How this grouping concept is used (e.g. hierarchical or non-hierarchical) depends on the originating system and cannot be checked by schema validation. The proposed code lists for *class* and *function* of a *CityObjectGroup* are incomplete; they only define *building separation* and *assembly* (*class*) and the definition of storeys (*function*).

Another structuring concept, only available for buildings and bridges, is the address. As CityGML is using the Extensible Address Language (xAL) schema from OASIS, all features from xAL can be used to structure the model.

Problem

The *CityObjectGroup* is a very general grouping concept and cannot uniquely represent specific (hierarchical or non-hierarchical) spatial structures in cities. This must be left to the originating system of the data and can only be checked with additional effort. The definition of the attributes *class* and *function* is very vague and the defined code lists values are incomplete.

Creating a spatial structure of a city by using the address can only be applied to buildings and bridges. Other features like vegetation or city furniture, which have no addresses, cannot be structured in such a way.

Recommendations

It is recommended to review the *CityObjectGroup* concept, if it is suitable for creating spatial structures of a city.

In general, it would be more reasonable to rethink the spatial structure concept of CityGML in total and to develop a concept, which can be used consistently for hierarchical and non-hierarchical spatial structures in a city model.

3.4.7 Representation of the spatial structure of a building

Motivation

The different rooms of a building are typically integrated into one or more spatial structures. For vertically structuring a building, the concept of floors (or storeys) is frequently used. Furthermore, many applications require additional aggregation concepts for rooms or building elements, e.g. to represent property rights (apartments) or parts of a building with homogeneous thermodynamic conditions (thermal zones).

Current state

CityGML has no specific concepts for spatially structuring the interior of a building. There is a general grouping concept (*CityObjectGroup*) for arbitrary city objects, which principally could be used to aggregate *Room* objects to storeys, apartments or thermal zones. By attributive information, it is possible to specify the number of storeys and the height of the different storeys, but a *Room* object cannot be related with a selected storey number.

Problem

For many simulation applications, the general CityGML grouping concept and the attributive representation of storey numbers and storey heights are not sufficient. The general *CityObjectGroup* class is difficult to process and lacks properties with specific semantics like, e.g. a storey number or an own geometric representation of the aggregation.

Recommendations

Define two new CityGML features:

- *Storey* for the representation of building storeys. The *Storey* class has specific properties (storey number, height, classification, and geometrical representations in correspondence with the new LoD concept) and optionally aggregates *Room*, *_BoundarySurface* and *BuildingInstallation* objects.
- *BuildingUnit* for arbitrary aggregations of *Room* and *BuildingInstallation* objects. Like the *Storey* class, *BuildingUnit* needs a number of specific attributes, including explicit geometrical representations.

3.4.8 Modelling of Utility Networks

Motivation

The STREAMER project revealed that for planning new or majorly renovated buildings it is highly important to take into account the neighbourhood of the planning object. This neighbourhood not only consists of other buildings and traffic infrastructure, but also the network infrastructure for supply with energy (electricity, gas, district heating), fresh water and sanitary water must be considered.

Current state

The actual version 2.0 of the CityGML standard contains no concepts for modelling utility networks. Based on developments of the TU Berlin [Becker2012], a draft of a CityGML extension (Utility Networks ADE) exists, which enables to topologically, geometrically and technically represent arbitrary utility networks.

Problem

The existing prototype of the Utility Networks ADE was developed to support one specific use case: The simulation of intersectorial cascading effects in the failure of critical infrastructures. For supporting other use cases in the context of STREAMER like planning processes, the data model needs to be revised and extended. Furthermore, in order to be supported by commercial software products, the extension should be integral part of the base standard CityGML in the next version.

Recommendations

Rework and extend the Utility Networks prototype according to the STREAMER needs, and contact the responsible OGC committees (CityGML Standard Working Group) to integrate the extension into the next version 3.0 of the base standard.

3.4.9 Modelling of physical materials

Motivation

For simulating the behaviour of complex physical systems, many physical parameters need to be known. For simulations of buildings this especially means, that physical parameters describing the energetic performance of façade, roof, and slab materials (e.g. density, thermal conductivity, specific heat) must be available.

Current state

In the CityGML standard, only the visual appearance (colour, texture) of, e.g., wall or roof surfaces can be represented, but not the physical parameters of the corresponding building elements. With the so-called Energy ADE [EnergyADE2017], an extension of the standard is available, including among others a concept for building materials and multi-layered constructions.

Problem

The building material model is part of a CityGML extension, and no integral component of the base standard. The model is restricted to layered constructions, composed of solid or gaseous materials. Liquid materials, which are important to physically describe the commodities in utility networks, are not regarded so far.

Recommendations

The material part of the CityGML Energy ADE should be extended and transferred into the next version 3.0 of the base standard.

3.4.10 Additional properties for CityGML class *Building* / *BuildingPart*

Motivation

In the property set *Pset_BuildingCommon*, the IFC standard defines a number of general properties on building level.

Current state

Only a few of the properties contained in the IFC property set *Pset_BuildingCommon* are also available in the CityGML classes *Building* and *BuildingPart*

Problem

It could be a problem for certain applications that some high-level building properties are not available in CityGML.

Recommendations

Check all properties of the IFC property set *Pset_BuildingCommon* for integration into CityGML

Pset_BuildingCommon:

- *Reference*
- *BuildingID*
- *IsPermanentID*
- *ConstructionMethod*
- *FireProtectionClass*
- *SprinklerProtectionAutomatic*
- *GrossPlannedArea*
- *NetPlannedArea*
- *NumberOfStoreys*
- *YearOfConstruction*
- *YearOfLastRefurbishment*
- *IsLandmarked*

3.4.11 Properties for CityGML class *_BoundarySurface*

Motivation

The abstract class *_BoundarySurface* is superclass for all boundary surfaces like wall -, roof – and ground surface. As these surfaces are representing both the outer shell of a building and the shell of a room, it is desirable to describe the shell in detail by properties of the boundary surfaces.

Current state

Currently, the class *_BoundarySurface* and all derived classes have no CityGML standard properties. There is only the possibility to add generic attributes to the boundary surfaces.

Problem

Using generic attributes allows only a data exchange when the attributes are informal. As they are not semantically described, the use of these attributes is very limited.

Recommendations

It should be discussed, which properties are desirable for boundary surfaces on city level. Properties of IFC building elements, like “Fire Rating”, “Combustible”, “Thermal Transmittance” and “Load Bearing”, are a good starting point for such a discussion.

3.4.12 Quantities

Motivation

For many applications on city level, quantities are relevant input parameters. Basic quantities like length, area and volume might be possible to be derived from the geometry. As soon as the quantities are not reflected in the geometry, they must be given as properties.

Current state

There are some quantities / measurements in CityGML available. In the building and bridge module, the *measuredHeight* indicates the height of the building, in the vegetation module, the *height*, the *trunkDiameter* and the *crownDiameter* specify the vegetation object. Different measures like minimum and maximum or net and gross of the same quantity are not possible. Additional information about the quantities are also not provided.

Problem

Many quantities cannot be taken from the geometry. Especially, when using different levels of detail the quantities calculated on base of these geometries can vary.
Quantities must not always follow the geometry of the model. For example, the building volume might be subdivided into a volume of the building body and the volume of the building roof. It is common sense to have net and gross measures for geographic features.

Recommendations

It is recommended to discuss a general concept for quantities. Quantities can represent a resource in the sense of chapter 3.4.4, which can be used by all thematic domains. It is also recommended to discuss a concept to specify the quantities in detail (accuracy, tolerances, reference etc.).

3.4.13 Placemark / Bookmark

Motivation

A city model consists not only of physical objects. There is also the need to store “virtual” features in the model. Such “virtual” features might be a viewpoint, a camera path, a document or an annotation.

Current state

Currently, there is no possibility to represent something, which is only virtually linked to a certain place on earth.

The generic city object is introduced to cover objects, which are not represented in one of the modules. As a very general concept, it is not designed to handle features like viewpoints.

Problem

In order to really use “Virtual” features like viewpoint or annotations, the concept of generic city object is much to general. Having only *class*, *function* and *usage* as standard properties, it is not able to represent the semantic and the properties of e.g. viewpoints or annotations.

Recommendations

It is recommended to think about a concept to represent such “virtual” features. This includes not only the data modelling but also the possible implementation in originating and importing applications.

4. Summary and Conclusion

The STREAMER workflow relies on the usage of standardised data models. On building level, the buildingSMART standard Industry Foundation Classes (IFC) is most important. IFC is the backbone of the STREAMER workflow, which has been intensively used by several partners during the complete project duration. Starting with IFC version 2x3, at the end of STREAMER the new version IFC 4 is used for the complete process chain. Therefore, all IFC related recommendations refer to this version of the standard. Besides general recommendations (e.g. coordinate reference systems, owner history, model harmonisation), some specific recommendations to support energy simulations with IFC are given (e.g. property sets for energy demand and consumption). Further deficiencies became obvious with using the IFC technology in the application area of healthcare districts (indoor air quality), and in the early design phase (program of requirement). For the life cycle assessment of buildings the capability of IFC was compared to the Life Cycle Data Network of the European community, which resulted in further recommendations for improving IFC.

For improving the quality of the generated building information models and supporting model checking, the concept of Model View Definition is deployed. Model View Definitions are an important method to control the data flow between certain process steps. For the formal description of a Model View Definition, buildingSMART has developed the mvdXML format. In STREAMER, several mvdXML based view definitions have been defined. The corresponding experiences resulted in a number of recommendations for both versions 1.1 and version 2.0 of mvdXML, which are specified in this deliverable.

STREAMER is a European project with partners from different countries with different native languages. Therefore, it is highly important to get a common understanding of all IFC entities and property values in combination with these different languages, which is principally supported by the buildingSMART data dictionary (bSDD) standard. Unfortunately, bsDD still is not covering all the needs of BIM, and therefore only a general recommendation for including also proprietary attributes and classification can be stated.

STREAMER is not only focusing on single buildings in detail, but also considers the building's neighbourhood including the district level. In order to cover this level, the OGC CityGML standard is used. As the CityGML base model only covers basic information on building and infrastructure objects, it is necessary to extend it by using the Application Domain Extension mechanism. Beside the recommendation to intensify the development of the both existing extensions for energy calculations of building (Energy ADE) and utility networks (UtilityNetworks ADE), also some recommendation for improving the base standard are formulated. This includes the improvement of the documentation, a restructuring of the CityGML modules, and a number of proposals for new features and properties.

The use of semantic web technology (e.g. ifcOWL, PMO) within STREAMER was tested in the first phase of the project. As no real advantages in using this technology for the STREAMER workflow were realised, the technology was not used and therefore no recommendations to the corresponding standardisation organisation W3C can be given.

As the STREAMER consortium is well represented in both buildingSMART International (MSG, ISG) and the Open Geospatial Consortium (CityGML SWG), most of the recommendations have already been

communicated to the corresponding organisations and working groups. Some issues are already considered in the current developer version of IFC 5.

5. References

[AIA2007]	AIA Best Practices: Defining the Architect's Basic Services, American Institute of Architects, BP 15.01.01, http://www.aia.org/aiaucmp/groups/secure/documents/pdf/aiap026834.pdf, July 2007
[Becker2012]	Becker, Thomas; Nagel, Claus; Kolbe, Thomas H. (2012) Semantic 3D modelling of multi-utility networks in cities for analysis and 3D visualization. http://mediatum.ub.tum.de/doc/1145724/287720.pdf utility networks in cities for analysis and 3D visualization:
[Benner2013]	Joachim Benner, Andreas Geiger, Gerhard Gröger, Karl-Heinz Häfele, Marc-Oliver Löwner: ENHANCED LOD CONCEPTS FOR VIRTUAL 3D CITY MODELS. ISPRS Annals of the Photogrammetry / Remote Sensing and Spatial Information Sciences, II-2/W1, pp. 51 – 61, 2013.
[BIM2016]	National BIM Standard – United States, web site, Frequently asked Questions "What is BIM", https://www.nationalbimstandard.org/faqs#faq1, last access June 2017
[BS 8541-6]	BS 8541-6:2015, Library objects for architecture, engineering and construction. Product and facility declarations. Code of practice, 2015
[bSIabout2016]	buildingSMART International web site, http://buildingsmart.org/about/, last access December 2016
[bSIcore2016]	buildingSMART International web site, http://buildingsmart.org/about/operating-vision/core-programs/, last access December 2016
[bSIDD2016]	buildingSMART International web site, http://buildingsmart.org/standards/standards-library-tools-services/data-dictionary/, last access April 2017
[bSIopenBIM2017]	buildingSMART International web site, http://buildingsmart.org/standards/technical-vision/, last access June 2017
[bSIISG2017]	buildingSMART International web site, http://www.buildingsmart-tech.org/about-us/isg, last access June 2017
[bSIMSG2017]	buildingSMART International web site, http://www.buildingsmart-tech.org/about-us/msg, last access June 2017
[bSIorgans2016]	buildingSMART International web site, http://buildingsmart.org/about/organization/, last access December 2016
[CityGML2012]	Gröger, G., Kolbe, T. H., Nagel, C., Häfele, K.-H.: OGC City Geography Markup Language (CityGML) Encoding Standard, Version: 2.0.0, Open Geospatial Consortium, OGC 12-019, 2012
[CityGMLGitHub]	CityGML development, https://github.com/opengeospatial/CityGML-3.0, last access March 2017

[CityGMLSWG] CityGML Standards Working Group, http://www.opengeospatial.org/projects/groups/swg last access March 2017.

[Defacto2016] De facto standard, Wikipedia, https://en.wikipedia.org/wiki/De_facto_standard, last modification 17th July 2016, last access December 2016

[EnergyADE2017] CityGML Energy Application Domain Extension (ADE), https://github.com/cstb/citygml-energy. Last access February 2017

[EPSG2016] EPSG Geodetic Parameter Registry, Version: 8.9.5, http://www.epsg-registry.org/, last access October 2016

[IFC4Add2] IFC 4 – Addendum 2 online documentation, http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/, © buildingSMART 1996-2016, last access December 2016

[IFC4Add2-1] IFC 4 – Addendum 2 online documentation for Pset_SpaceCoveringRequirements, http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/schema/ifcproductextension/pset/pset_spacecoveringrequirements.htm, © buildingSMART 1996-2016, last access December 2016

[IFC4Add2-2] IFC 4 – Addendum 2 online documentation for IfcExternalReference, http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/schema/ifcexternalreferenceresource/lexical/ifcexternalreference.htm

[IFC4DTView2016] buildingSMART International technical web site, IFC4 Design Transfer View, http://www.buildingsmart-tech.org/specifications/ifc-view-definition/ifc4-design-transfer-view/ifc4-design-transfer-view, 2016, last access December 2016

[IFC4RefView2017] buildingSMART International technical web site, IFC4 Reference View, http://www.buildingsmart-tech.org/specifications/ifc-view-definition/ifc4-reference-view/ifc4-reference-view, last access June 2017

[ILCD2014] Developer support, http://eplca.jrc.ec.europa.eu/ELCD3/developerPage.xhtml;jsessionid=2703835E85D137D1E5882A56E8BBD0C6, last access February 2017

[ISO639] ISO 636, Codes for the representation of names of languages, 2002

[ISO10303-11] ISO 10303-11:2004, Industrial automation systems and integration -- Product data representation and exchange -- Part 11: Description methods: The EXPRESS language reference manual, 2004

[ISO12006-3] ISO 12006-3:2007, Building construction -- Organization of information about construction works -- Part 3: Framework for object-oriented information, 2007-04, 2007

[ISO14040] ISO 14040:1997, Environmental management - Life cycle assessment - Principles and framework, first edition 1997-06-15, 1997

[ISO16739]	ISO 16739:2013, Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries, 2013
[ISO19101-1]	ISO 19101-1:2014, Geographic information — Reference model — Part 1: Fundamentals, 2014
[ISO19136]	ISO 19136:2007, Geographic information – Geography Markup Language (GML), 2007
[ISO29481-1]	ISO 29481-1:2016, Building information models -- Information delivery manual -- Part 1: Methodology and format, 2016
[ISO29481-2]	ISO 29481-2:2012, Building information models -- Information delivery manual -- Part 2: Interaction framework, 2012
[ISOStandard2016]	ISO web site, http://www.iso.org/iso/home/standards.htm, last access December 2016
[OGCvision2016]	Open Geospatial Consortium
[OGCWikipedia2016]	Open Geospatial Consortium, Wikipedia, https://en.wikipedia.org/wiki/Open_Geospatial_Consortium, last modification 20th September 2016, last access December 2016
[SIG3D2017]	SIG3D – Special Interest Group 3D, http://www.sig3d.org/, last access March 2017
[Standardisation2016]	Standardisation, Wikipedia, https://en.wikipedia.org/wiki/Standardization, last modification 02nd December 2016, last access December 2016
[TMY2017]	Typical meteorological year, Wikipedia, https://en.wikipedia.org/wiki/Typical_meteorological_year, last modification 14th February 2017, last access April 2017
[WebService2016]	Web service, Wikipedia, https://en.wikipedia.org/wiki/Web_service, last modification 02nd October 2016, last access October 2016

6. Appendix

6.1 Complete list of buildingSMART standards

Standard	Long Name	Scope	Remarks
IDM	Information Delivery Manual	Processes	ISO Standard
IFC 4	Industry Foundation Classes Version 4	Building	ISO Standard
BCF XML	BIM Collaboration Format XML	Collaboration, Change Coordination	Pre-release
BCF API	BIM Collaboration Format REST API		Pre-release
IFD	International Framework for Dictionaries	Mapping of Terms	
mvdXML	Model View Definition XML	Formal description of Model View Definition	
Design Transfer View	IFC 4 Design Transfer View	Model View Definition to hand over models to perform in next work flows, allowing modifications of its content	
Reference View	IFC 4 Reference View	Model View Definition to hand over models for downstream applications, which usually don't perform modifications	
Infrastructure Alignment	IFC 4 Infrastructure Alignment	Model View Definition to hand over 3D and 2D alignment information for spatial location of infrastructure assets	

6.2 Complete list of OGC standards

Complete List of OGC standards (September 2016)

Standard	Long Name	Scope	Remarks
ARML2.0	Augmented Reality Markup Language	Interchange format for Augmented Reality (AR) applications to describe an AR scene	XML
Cat	Catalogue Services Standard 2.0 Extension Package for ebRIM Application Profile: Earth Observation Products		
Catalogue Service	Catalogue Service		
CityGML	City Geography Markup Language	Format for the storage and exchange of virtual 3D city models	XML
Coordinate Transformation	Coordinate Transformation Service		
Filter Encoding	Filter Encoding		
GML in JPEG 2000	GML in JPEG 2000 for Geographic Imagery Encoding		
GeoSparql	GeoSPARQL - A Geographic Query Language for RDF Data		
GML	Geography Markup Language	Grammar for expressing geographical features	ISO Standard
GeoXACML	Geospatial eXtensible Access Control Markup Language		
IndoorGML	Indoor Geography Markup Language	open data model and XML schema for indoor spatial information	
KML	formerly Keyhole Markup Language	KML is a language focused on geographic visualisation, including annotation of maps and images	

OpenLS	Location Service		
Moving Features	Moving Features		
NetCDF	network Common Data Form		
O&M	Observations and Measurements		
OpenGeoSMS	Open Geo Short Message Service		
OpenMI	Open Modelling Interface		
OpenSearchGeo	OpenSearch Geo and Time Extensions		
OWS Context	OGC Web Services Context Document		
PubSub	Publish/Subscribe Interface		
PUCK	PUCK Protocol Standard		
SWE Common Data Model	Sensor Web Enablement Common Data Model		
SWE Service Model	Sensor Web Enablement Service Model		
SensorML	Sensor Model Language		
SOS	Sensor Observation Service		
SPS	Sensor Planning Service		
SensorThings			
Simple Features	Simple Feature Access		
Simple Features CORBA	Simple Features for CORBA		
Simple Features OLE/COM	Simple Features for OLE/COM		
Simple Features SQL	Simple Feature Access: SQL Option		
SLD	Styled Layer Descriptor		
Symbology Encoding	Symbology Encoding		
TJS	Table Joining Service		

WaterML	Water Markup Language		
	Web Coverage Processing Service		
WCS	Web Coverage Service		
WFS	Web Feature Service		
	Web Map Context		
WMS	Web Map Service		
WMTS	Web Map Tile Service		
WPS	Web Processing Service		
	Web Service Common		
WKT CRS	Well-known text representation of coordinate reference systems		