Change toolkit for digital building permit

Deliverable number	D2.2
Deliverable name	CHEK IFC specification
Work package number	WP2 Information requirements for the DBP use case
Deliverable leader	RDF Ltd.
Dissemination Level	Public

Status	Draft
Version Number	V 1.0
Due date	M18
Submission date	28/03/2024

Project no. 101058559

Start date of project: 1 October 2022

Duration: 36 months

File name: CHEK_101058559_D2.2-CHEK_IFC_specification_V1.0-FINAL

Funded by the European Union

This project has received funding from the European Union under the Horizon Europe Research & Innovation Programme 2021-2027 (grant agreement no. 101058559).

Authors and contributors

Author	Organisation	E-mail
Peter Bonsma	RDF	peter.bonsma@rdf.bg
Iveta Bonsma	RDF	iveta.bonsma@rdf.bg

Quality control

Author	Organisation	Role	Date
Léon van Berlo	buildingSMART	Reviewer	18/03/2024
Silvia Mastrolembo Ventura	UBS	WP leader	27/03/2027
Francesca Noardo	OGC	Coordinator	28/03/2028

Document history

Release	Description	Date	Author
V0.1	Draft	01/03/2024	Peter Bonsma
V0.2	Draft	06/03/2024	Peter Bonsma
V0.3	Draft	11/03/2024	Peter Bonsma
V0.9	Draft	26/03/2024	Peter Bonsma
V0.91	Draft	27/03/2024	Peter Bonsma
V1.0	Final	28/3/2024	Peter Bonsma

Contents

1. Executive Summary

D2.2 delivers a solution that makes sure building information models are rich enough to allow execution of rules in WP4. This solution is built in a computer-interpretable format, following open standards and defined as CHEK IFC. Not only is a computer-interpretable format required, but open standards should be used, and the solution must be relevant and useful to designers and municipalities.

CHEK is not the only project looking to define similar knowledge and together with the fact that the most obvious technology at the moment of writing the proposal, i.e. mvdXML, was withdrawn by buildingSMART International (bSI), the first logical step was to apply a state-of-the-art research on relevant open standards that could be used by CHEK to specify CHEK IFC.

From the state-of-the-art research in collaboration with all partners in WP2 and several technical specialists from the advisory board, the eventual choice was made to use IDS 1.0 in combination with micro-services for managing the geometrical algorithms of the CHEK IFC specification. Note that IDS 1.0 is foreseen to become the official standard, currently not yet available. Nonetheless the most recent draft of IDS is available and having bSI as a partner allowed us to understand the foreseen functionality of IDS 1.0 and wide adoption in the market of the draft version of this open standard already.

Together with the software vendors supporting BIM (DiRoots, CYPE and Xinaps) and designers within CHEK the best practical choice for IFC4 ADD2 TC1 (ISO 16739-1:2018) was made. Knowing that more recent versions of IFC have benefits when combining BIM and GIS especially on the geo-referencing part.

As selection of an IFC schema is more or less a theoretical exercise it is required to define a Model View Definition that can be seen as a set of agreements from software vendors about what parts of a specific IFC schema to use and how to populate it. The final choice for CHEK IFC has become:

IFC4 ADD2 TC1, Reference View MVD + second level space boundary

With this basis, it is possible to use the information requirements for the digital building permit use case as gathered in D2.1 to identify the gaps between what is already available in the IFC files exported from BIM authoring platforms and what is required to execute the rules as implemented in WP4 – Software development. The gaps identified are written in IDS, where possible following standard IFC classification and property sets and where required through a CHEK specific property set (defined as PSD – Property Sets Definition, an open standard for defining PSet's). Both the IDS file and PSD file are made available and are the basis of the specification of CHEK IFC.

A proper understanding and development of CHEK IFC are not possible without some key validation libraries able to validate any IFC file against above agreements and the created IDS file. A validation for IFC schema language (ISO 10303-11:2004) and a validation for IDS 0.9.6 have been created and embedded in a free to use viewer. The IDS validation has been developed as open source given that it requires a C-language SDAI library (ISO 10303-22).

To reduce the burden on designers for some of the alphanumeric numbers that are required to be filled in to ensure validity against CHEK IFC, a small set of micro-services has been created.

With this deliverable, a first version of CHEK IFC has been defined. A proper computer interpretable solution allowing validation of BIM data to ensure it is rich enough to be input for the Digital Building Permit rules (T2.1) that must be executed in WP4. During this Task 2.3, it became clear CHEK IFC will not be a static solution during the second half of CHEK. However, the approach allows easy and properly validated extension and adjustment of CHEK IFC.

2. Introduction

The CHEK project enables support for Digital Building Permit processes. Here the Digital Building Permit rules are based on both BIM and GIS data. The CHEK project focusses on use of open standards and concerning the source data specifically IFC for BIM and CityGML for GIS data. Within the objectives of CHEK, and WP2 – Information requirements for the DBP use case specifically, there is the interpretation of formalization of building permit regulations as input requirements to checking software and to specify the regulatory information requirements (building and city models) in standard-compliant formats (IFC and CityGML).

This deliverable focusses on how BIM data should be delivered to enable execution of Digital Building Permit rules related to and/or based on BIM data. The solution has to be defined in a computer-interpretable format, allowing the compliance checking software (WP4) to read regulations and related parameters automatically. Here the project proposal defines IFC as the open standard used to represent building-related data. This deliverable (D2.2 as a result of Task 2.3) defines CHEK IFC specification, the underlying standards and a solution to handle CHEK IFC.

Within WP2 of CHEK the next deliverable (D2.3 as a result of Task 2.4) will answer a similar question for GIS, i.e. CHEK CityGML. Note that both CHEK IFC and CHEK CityGML specifications are very relevant for the converters from CityGML towards IFC and IFC towards CityGML as main results of WP3. The converters will not only depend on the base choices made within this D2.2 and D2.3, but it will also use CHEK IFC and CHEK CityGML as main content for input. Therefore, choices made within this deliverable, like the base IFC schema (and base CityGML version) will influence the work in WP3 as these schema versions have to be supported at least.

The actual content of CHEK IFC represents the requirements as defined in D2.1 - Regulations interpretation and needs identification for CHEK DBP (submitted on M13), which describes the methodology adopted from the interpretation of building permit regulations (T2.1) to the definition of information requirements (T2.2). The later Task 2.5 will focus on development of software able to validate the compliance of design proposals with CHEK IFC (D2.2) and CHEK CityGML (D2.3) specifications (i.e., formal validation – see D1.1) Once validated for compliance with open standard specifications, BIM-based design proposal will be then checked for compliance with in-force building permit regulations. The actual implementation of the code checking rules will be applied in WP4 and the software solutions executing the rules will use CHEK IFC and / or CHEK CityGML as input.

In practical sense, the goal of CHEK IFC and this deliverable is therefore to have ‘rich’ enough DBP purpose-oriented BIM data to enable (automatic) execution of the Digital Building Permit rules in WP4. By rich we mean that enough relevant data is available, rich in the sense of a clear enough semantic meaning.

2.1 Objective

As described above the objective is that we require rich enough DBP purpose-oriented BIM data to execute the Digital Building Permit rules. CHEK IFC is a specification of BIM data enabling validation if the delivered BIM data is exactly that.

Make all digital data and services required - i.e. 3D city models, zoning plans, BIM, monitoring information, such as weather information - 100% interoperable (human and machine to machine) within all the software needed (authoring, analysis, conversion, communication, viewers tools), meaning that they can be consistently read, used and exported multiple times by the software and likely converted into different formats without any loss of the required information. The focus and demonstration will regard the CHEK regulations.

Part of the objective is to use open standards to reach our goal of enforcing data and service interoperability. Next to that the solution needs to be practical and future proof. The CHEK IFC specification should not only follow open standards but should allow designers to create files valid against the CHEK IFC specification. Furthermore, the solution needs to be useful during the project and afterwards; this is for CHEK IFC an important objective as the landscape in relevant open standards changed considerably over the past years. As we will learn important assumptions during the proposal phase are not valid anymore during this task and we base CHEK IFC on open standards that did not yet reach their first official standard release when writing this deliverable.

3. Technology

From a technical perspective nothing new, some still active rules date back to the early STEP days, years before IFC started development (i.e. ,1994). Also, several organizations and companies are famous for their work on defining and checking rules, examples here are Solibri (currently owned by Nemetschek) and the FZK viewer of Karlsruhe Institute of Technology.

For the past 10 years technologies have been water-clear; there was only one technology of choice, i.e. mvdXML with a limited set of versions and supported by solutions like IfcDoc.

Recently, the situation has changed; bSI deprecated mvdXML and introduced new standards like IDS. For CHEK it is very important to take into account the new wave from bSI and make a technology choice that allows results to be used after CHEK also.

3.1 Alternatives

As presented in the first physical meeting for CHEK held in Delft (Month 4) here the five options as currently can be identified:

- mvdXML (https://technical.buildingsmart.org/standards/ifc/mvd/mvdxml/)
- IDS (https://technical.buildingsmart.org/projects/information-delivery-specification-ids/)
- Gherkin, an ordinary language parser (https://en.wikipedia.org/wiki/Cucumber_(software)#Gherkin_language)
- EXPRESS (https://en.wikipedia.org/wiki/EXPRESS_(data_modeling_language)
- Other alternatives

3.1.1 mvdXML

From a historical point of view mvdXML is still the most logical choice at the moment, for this reason it was also the proposed technology in the CHEK proposal. However, as we are developing a 'CHEK IFC' solution, IFC is developed by bSI and bSI is part of the project the fact that bSI deprecated mvdXML we should not use mvdXML as a base for CHEK IFC. Technically mvdXML is not computer interpretable, since a special string parser needs to be developed to interpret the rules. The latest version of mvdXML did not follow the bSI consensus process, and therefore has no support from major implementors. There is also no documentation available from this new version. This leads to a situation where mvdXML is unimplementable. Large stakeholders and vendors have made it clear they don't see a path forward for mvdXML and therefore started the development of the Information Delivery Specification (IDS).

3.1.2 IDS 1.0

Information Delivery Specification (IDS) is a bSI standard defined as a computer interpretable document that defines the Exchange Requirements of model-based exchange. It is designed as a free, lightweight, standardized approach to information requirement checking.

If 'CHEK IFC' can be defined through IDS this would be a good option. Even though the standard is new, it is already implemented in a multitude of tools (including open-source ones). Since IDS is lightweight, it needs to be validated if the rules and knowledge that need to be modelled within CHEK IFC can be covered by IDS; this means is IDS expressive enough to allow us to define what is required to be present in IFC.

3.1.3 Gherkin

One of the great features of Gherkin is the freedom in defining rules and languages. This is however also one of its pitfalls, the fact that the language has so much freedom allows the user to write virtually any rule without restrictions in semantic meaning; the real understanding of the meaning of the rule is captured in source code.

bSI is using Gherking language for an open-source validation tool for IFC. The rules written in Gherkin in the validation service are generally true for all users of IFC since the validation service is a generic IFC validation. bSI has added a further limitation to this technology, i.e. the use of Python as preferred language (Gherkin itself does not limit a specific programming language).

The fact that the solution Gherkin itself solves the 'semantic meaning' within source code and the added restriction from bSI makes Gherkin not a preferred option for CHEK IFC.

3.1.4 EXPRESS (ISO 10303-11:2004)

ISO 10303 specifies a language by which aspects of product data can be defined. The language is called EXPRESS. EXPRESS is a data specification language as defined in ISO 10303-1. It consists of language elements that allow an unambiguous data definition and specification of constraints on the data defined.

Until now all IFC versions (incl. IFC 4.3 which is published by ISO and IFC 4.4 in development) have been published as EXPRESS. It defines the schema and a complex set of WHERE rules and functions. The IFC standard itself is, however, much more than just the EXPRESS schema.

The technology is somewhat dated but expressive and well defined. Even very complex rules can be covered by ISO 10303-11:2004. It is expected that most rules needed for CHEK IFC could be covered.

Recently we have seen a growth in the number of tools and toolboxes supporting this specification, currently 5 out of the 16 available toolboxes for IFC support (incl. geometry) have a more-or-less complete support for ISO 10303-11:2004.

3.1.5 Alternatives (SPARQL, SHACL, …)

There are many alternatives available; if none of the above solutions would provide a proper solution for CHEK IFC. Especially SPARQL and SHACL from the Semantic Web domain offer attractive functionality. A separate meeting with Semantic Web specialists from sister project ACCORD was held.

3.2 Decision

The decision was made to select the open standard IDS 1.0 (which is currently published as 0.9.6). This even though the expressiveness of this open standard is relatively low and will be most probably official in the summer of 2024. The main reasons behind the choice are the already wide acceptance and adoption of this open standard. The simplicity allowed many software vendors to implement it in a short time and at the same time domain experts understand its purpose and use allowing a quick and fairly wide adoption in this early stage.

As expected, the low expressiveness will create several cases where none of the solutions listed above alone will cover the requirements that will be necessary to define and an extension with micro-services is proposed. The alternatives as well as their foreseen benefits and drawbacks have been discussed with all partners active in WP2, also two highly skilled specialists, i.e. Thomas Krijnen from the advisory board and André Borrmann, have joined the discussion and dedicated meetings with both have been held.

4. Tooling

Within the project proposal CHEK IFC has been defined more as a specification and not directly a set of software solutions. Nonetheless the fact it was decided to use the brand-new IDS standard (actually not even completed and finished at moment of writing this deliverable) and the consistency of the IFC file itself is of essence for further processing introduced the need to develop validation for both IFC and IDS.

Although Task 2.5 is foreseen to validate the actual CHEK IFC and CHEK CityGML specification with tooling developing a proper and brand-new open standard can only be done properly when there is tooling to test and validate the specification during development of the CHEK IFC specification itself. This is especially true for the complex IFC schema and new open standard IDS. Also, the foreseen changes in CHEK IFC make tooling essential. Here we focus on the most necessary tooling for developing the specification. Proper validation of IFC files following the specification is still foreseen as part of task 2.5.

4.1 Part of D2.2

To enable a proper definition of CHEK IFC that itself is strongly depending on the IFC schema definition and IDS it is essential for both understanding and feedback that validation of the IFC (EXPRESS) schema language (ISO 10303-11:2004) and validation of IDS is available. Therefore, as part of D2.2 the following tooling is developed.

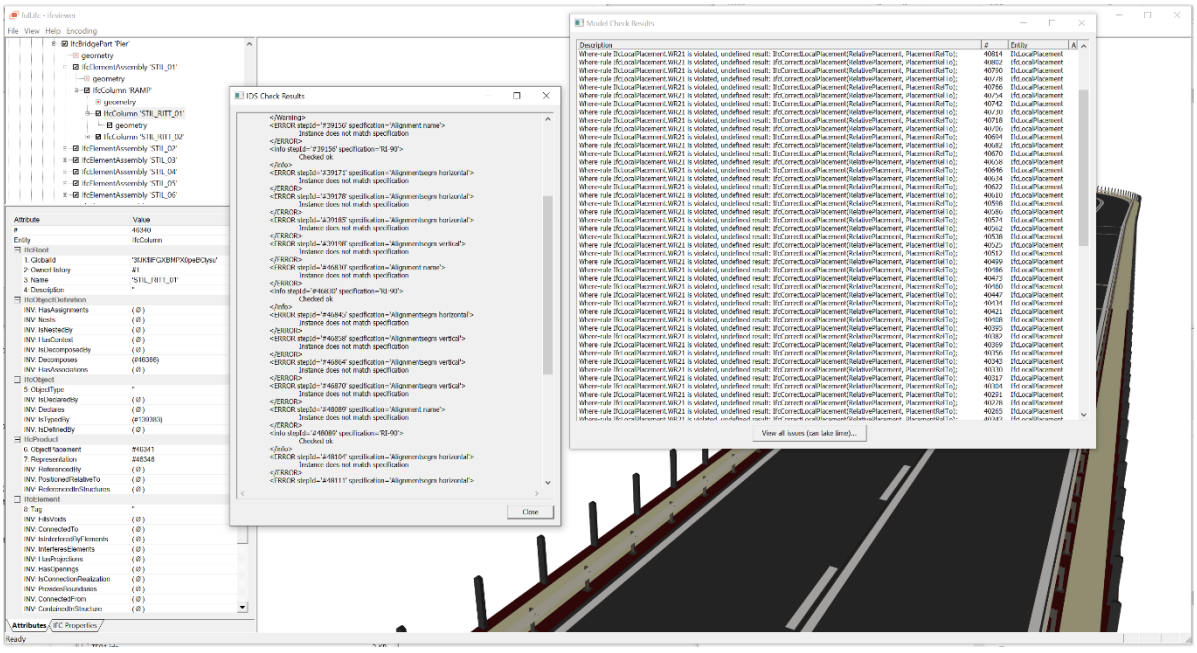

Figure 1 IFC Viewer with both validators for D2.2 embedded

4.1.1 EXPRESS Schema Validation (ISO 10303-11:2004)

Without proper IFC files there is no sense in using CHEK IFC; therefore, the first task required was to develop an IFC schema validation, at least for the schema used by CHEK IFC. A large set of files from vendors have been checked with this solution and issues have been sent to the developers of these tools so they can improve their solution by exporting IFC files.

Development of EXPRESS schema validation is not a trivial task. Although the language is an ISO standard (ISO 10303-11:2004) and has existed for almost 20 years it is not consistent and similar statements can lead to different results in different solution implementing ISO 10303-11:2004 to the letter. Even the very widely used C++ standard 11 (ISO/IEC 14882:2011) has similar issues where different compilers produce code generating different results on the same 100% correct C++11 source code. A small example is the following statement 'a[i++] = b[i++ + i] + c - i++;' where the language does not specify exactly the order of increasement of value i in the procedural statement.

In all sources developed for CHEK the standard C++11 will be used; however, the code will be written in a way that every well-known compiler (GCC, Visual Studio, Borland, XCODE) will produce the same solution.

Next to this the development of ISO 10303-11:2004 is far from a trivial task because of its expressiveness. Although this ISO specification is far away from a complete programming language the constructs and implementation of support for WHERE rules and FUNCTIONS requires implementation of an important part of a programming language. To enable support for one of the alternatives mentioned in chapter 2.2.3 the implementation was required to be implemented late-binding. This means that the software allows loading schema files during execution of the hosting application (where early-binding expects generation of source code for any specific schema and therefore recompilation of the solution with each change in the schema).

4.1.2 IDS 0.9.6

Validation of IDS 1.0 is close since the 0.9.6 version is expected to be renamed to 1.0 in the Summer of 2024. The current most used version in practice is IDS 0.9.6; a validator for this version has been implemented (called the ‘IDS Audit Tool’). The source code is developed as open source and made publicly available through a public GitHub page.

To enable the source code an EXPRESS library with an SDAI interface is required. The Standard Data Access Interface (SDAI) defines a low-level application programming interface to EXPRESS defined data, such as STEP Application Protocols. STEP defines the set of general SDAI operations in ISO 10303-22. These operations are then implemented in a specific programming language by a language binding, the pure C binding is followed in the source code. The source code is developed in C++11 (a version of the ISO/IEC 14882 standard for the C++ programming language).

4.2 Not part of D2.2 (part of WP2, i.e. Task 2.5)

The following tools are essential for CHEK in relation to CHEK IFC but are more relevant for validation of CHEK IFC files and less for defining the CHEK IFC specification itself. The development of these tools finds its logical place in Task 2.5.

4.2.1 IDS 1.0

As soon as IDS 1.0 becomes the official standard in the summer of 2024, the changes against the most used 0.9.6 will be implemented in CHECK.

4.2.2 PSD

The dedicated property set(s) for CHEK are defined in the official PSD (Property Set Definition). Although the PSD file is available for CHEK IFC there is no tooling to validate this PSD. Task 2.5 will cover development of a tool that checks the validity of the delivered PSD files.

Together with BSI it will be validated if PSD's could be integrated within the IDS 1.0 files.

4.2.3 Micro-Services

Several of the micro-services are developed as part of D2.2. They are especially relevant for population of properties, property sets and structures that are required for CHEK IFC, can be derived from the context, geometrical representation or otherwise and are a tedious task to ask the designer to fill in.

Within task 2.5 will be checked if the APIs of the micro-services can be standardized using Foundation API (https://github.com/buildingSMART/foundation-API) or Documents API (https://www.buildingsmart.org/documents-api-receives-final-standard-approval/).

For many of the micro-services, however, implementation of proper calculation in a generic manner is complex and time consuming, so much so that it cannot be executed in the time frame of CHEK and specifically Task 2.3. Often a lighter version of the implementation can be defined in micro-services that typically cover 70% to 80% of the cases. These micro-services provide an excellent method to validate user input from the designer; even though the delivered IFC file can be a 100% valid CHEK IFC file, the micro-services can identify potential discrepancies between alphanumeric values given and geometry available.

Note that micro-services can be used for validation of user-input and for generation of alphanumerical values to be stored in IFC. The proper use can vary, it could be that micro-services built for validation end-up being part of the CHEK IFC specification and fill in properties instead of validating user-input.

5. Requirements

The requirements define the actual parts of BIM and GIS data that are needed to be present in order to be able to prepare building information models appropriately and to execute the Digital Building Permit rules as will be done in WP4.

The main source for the information requirements for the DBP use-cases is D2.1 (M13 – T2.2). Four different categories of verification (i.e., distances, buildability index, building height and area of building spaces) were selected to be analysed. They were selected based on a discussion with municipalities and by confirming the priorities of these types of checks for the building permit use case also during an open event and an interaction with the community of practice held on the 12th of October 2023. The interpretation of 80 regulatory sentences was delivered in D2.1. Specifically, 32 normative articles were analysed for Ascoli Piceno (APC), 14 normative articles were analysed for Vila Nova de Gaia (GAI), for Lisbon 17 normative articles and, finally, 17 normative articles were analysed for Prague (IPR). In addition to the normative articles, the definitions of the entities that emerged during the textual interpretation were analysed.

There are additional inputs that could be of importance for the development of the CHEK IFC specification, i.e.:

- the requirements modelled by TUD / OGC in order to develop CHEK CityGML
- existing software capabilities by software vendors; for example, CYPE has an existing classification of building spaces (IfcSpace)
- requirements models from members of the community of practice; for example, the list offered and developed by Rotterdam (a large innovative city in the Netherlands)
- requirements from designers and municipalities that are missing in above list of sources
- The main activity is to translate the requirements into IDS.

In the context of IFC the classification is largely defined by the IFC schema. The IFC schema has a strict classification that is expected to be supported and understood by BIM authoring platforms. Classes that cannot be mapped to the IFCs require a special solution. A typical example here is the definition of the object “Facade”, identified as necessary for the four municipalities for checking distances and building height regulatory requirements. Within IFC a Facade (depending on its exact definition) normally is divided by splitted walls over different building stories. For this specific example a micro-service is developed grouping these walls through a concept called IfcGroup in IFC. Alternatively, when it is one object, it can be defined using an IfcClassificationReference.

Classes and properties within the defined level of information need for the DBP use case as defined in D2.1 mapped towards IFC:

Table 1 IDS for APC

Classification D2.1	IFC4 ADD2 TC1 entity	IDS + Micro-Services	Comment
Balcony / Porch / Terrace / Building Canopy	IfcSlab		
Building	IfcBuilding	requirement for one IfcBuilding instance requires the following property sets and properties: Chek_common.TypeOfConstruction Chek_common.Height Chek_common.IntendedUse Chek_common.IsCornerBuilding Chek_common.NumberOfBuildingLevels Chek_common.ResidentialType	
Building Level	IfcBuildingStorey	requirement for at least one IfcBuildingStorey instance	
Column	IfcColumn		
Door	IfcDoor		
Floor	IfcSlab		
Foundation	IfcFooting		
Parapet	IfcWall with IfcWallTypeEnum = .PARAPET.		
Space	IfcSpace		
Roof	IfcRoof (aggregation of IfcSlab)	requirement for Chek_common.RoofAngle for each IfcRoof instance	
Slab	IfcSlab		
Stairwell	IfcStair		
Wall	IfcWall	requirement for Pset_WallCommon with IsExternal	
Window	IfcWindow		

Table 2 IDS for GAIA

Classification D2.1	IFC4 ADD2 TC1 entity	IDS + Micro-Services	Comment
Alignment	IfcAnnotation	requirement for line representation for any IfcWall instance	IDS 1.0 will probably not yet support access to IfcShapeRepresentation
Building	IfcBuilding	requirement for one IfcBuilding instance requires the following property sets and properties: Chek_common.TypeOfConstruction Chek_common.Height Chek_common.IntendedUse Chek_common.IsCornerBuilding Chek_common.NumberOfBuildingLevels Chek_common.ResidentialType	
Building Level	IfcBuildingStorey	requirement for at least one IfcBuildingStorey instance	
Chimney	IfcChimney		
Facade	aggregation of IfcWall		Create IfcGroup as aggregation holder
Space	IfcSpace		
Slab	IfcSlab		
Wall	IfcWall	requirement for Pset_WallCommon with IsExternal	

Table 3 IDS for LISB

Classification D2.1	IFC4 ADD2 TC1 entity	IDS + Micro-Services	Comment
Alignment	IfcAnnotation	requirement for line representation for any IfcWall instance	IDS 1.0 will probably not yet support access to IfcShapeRepresentation
Building	IfcBuilding	requirement for one IfcBuilding instance requires the following property sets and properties: Chek_common.TypeOfConstruction Chek_common.Height Chek_common.IntendedUse Chek_common.IsCornerBuilding Chek_common.NumberOfBuildingLevels Chek_common.ResidentialType	
Building Level	IfcBuildingStorey	requirement for at least one IfcBuildingStorey instance	
Chimney	IfcChimney		
Facade	aggregation of IfcWall		Create IfcGroup as aggregation holder
Space	IfcSpace		
Slab	IfcSlab		
Wall	IfcWall	requirement for Pset_WallCommon with IsExternal	

Table 4 IDS for IPR

Classification D2.1	IFC4 ADD2 TC1 entity	IDS + Micro-Services	Comment
Balcony / Porch / Terrace / Building Canopy	IfcSlab		
Building	IfcBuilding	requirement for one IfcBuilding instance requires the following property sets and properties: Chek_common.TypeOfConstruction Chek_common.Height Chek_common.IntendedUse Chek_common.IsCornerBuilding Chek_common.NumberOfBuildingLevels Chek_common.ResidentialType	
Building Level	IfcBuildingStorey	requirement for at least one IfcBuildingStorey instance	
Ceiling	IfcCovering		
Eave Cornice	IfcRoof		
Gable	IfcWall	requirement for second level space boundaries	
Parapet	IfcWall with IfcWallTypeEnum = .PARAPET.		
Space	IfcSpace		
Roof	IfcRoof (aggregation of IfcSlab)	requirement for Chek_common.RoofAngle for each IfcRoof instance	
Slab	IfcSlab		
Wall	IfcWall	requirement for Pset_WallCommon with IsExternal	
Window	IfcWindow		

The properties and classes are embedded in dedicated CHEK IDS and micro-services. Together with the validation solution this defines CHEK IFC. It is expected that during the second half of the project CHEK IFC will be updated, and it is essential to have a flexible solution as initial IDS files require to be changed during implementation of the actual rules within WP4. For example, from M14 onwards, minor activities with the scope of T2.2 continued to revise and

validate the proposed methodology on a wider data set by integrating additional input from T2.1 (i.e., interpretation of further regulations). The interpretation work has been extended in order to obtain a broader range of verifications to be considered for establishing the information requirements and for implementation through the software solutions in WP4, starting from accessibility checks and additional urban indices. The additional input will have to be considered in future extension of CHEK IFC specification.

6. Specification

6.1 IFC Versions

There are many different versions from IFC available, here a limited list (excluding IFC 1.5, IFC 1.5.1 and earlier):

Table 5 IFC Versions

Version	Name	Published	Status
2.0.0.0	IFC 2.0	1999	Retired
2.1.0.0	IFC2x	2000	Retired
2.1.1.0	IFC2x ADD1	2001	Retired
2.2.0.0	IFC2x2	2003	Retired
2.2.1.0	IFC2x2 ADD1	2004	Retired
2.3.0.0	IFC2x3	2005	Retired
2.3.0.1	IFC2x3 TC1 (ISO/PAS 16739:2005)	2007	Official
4.0.0.0	IFC4 (ISO 16739:2013)	2013	Retired
4.0.1.0	IFC4 ADD1	2015	Retired
4.0.2.0	IFC4 ADD2	2016	Retired
4.0.2.1	IFC4 ADD2 TC1 (ISO 16739-1:2018)	2017	Official
4.1.0.0	IFC 4.1	2018	Withdrawn
4.2.0.0	IFC 4.2	2019	Withdrawn
4.3.0.0	IFC 4.3	2022	
4.3.1.0	IFC 4.3 ADD1	2023	
4.3.2.0	IFC 4.3 ADD2 (ISO 16739-2:2024)	2023	ISO / CEN approved
4.4 dev	IFC 4.4 - in development	202?	

Given the major improvements introduced with IFC4 (introduced a decade ago) and its majority in the market it makes perfect sense not to base CHEK IFC on versions before IFC4; this even though in practice many of the IFC files created and exchanged today are still following the older IFC2x3 TC1 schema.

Given the capabilities from applications used by the designers and the applications in development by the software vendors part of CHEK it is a bridge to far to use the very recent infrastructure version IFC4.3 ADD2 that has been accepted as an ISO / CEN standard 3rd of January 2024.

Although the schema has been available for a while already (roughly half a year) and several software vendors are actively implementing support for this version there is currently no (stable) support for designers even when using early beta releases. Also, not all software vendors within CHEK supporting IFC do have support for this version already. Although IFC 4.3 is mainly focused on extensions for infrastructure there are also very relevant improvements for CHEK. One of them is the new capability for geo-referencing an IFC file, allowing more rich semantics concerning

global placement of IFC objects that itself is closer to the way GIS standards like CityGML / CityJSON define geo-referencing.

Given the above restrictions and the fact that IFC 4.1 and IFC 4.2 are respectively retracted and never officially published the choice for IFC 4 seems clear. There are several revisions available from the IFC4 schema, however the most widely used version is IFC4 ADD2 TC1 released in 2018 (internally coded 4.2.1.0) and it received the official ISO certification.

Concluding IFC4 ADD2 TC1, i.e. ISO 16739-1:2018 is the base IFC standard schema for CHEK IFC. Note that in this time the schema identifiers were inconsistent in the way that several different schemas use the same schema identifier in the IFC file distributed. This means that IFC files created against other schemas could not be identified unless a real changed entity, attribute or rule was used. The schema validation as can be found in chapter 4 can validate the IFC file against this schema and in case of a violation also check against any other schema version to identify a probable cause for a violation if this was related to the use of an older or incorrect schema.

6.2 IFC MVD

The selection of the IFC ADD2 TC1 schema as a base for CHEK IFC allows designers to select the proper export settings within their CAD application. However, an IFC schema is a rather theoretical set of entities, attributes and rules allowing a wide variety of constructs to be created. Following full capability of an IFC schema would make import of IFC files very complex and proper certification of exporters and importers close to impossible. Therefore, on top of each schema implementers of the large CAD companies come together and agree on specific use of that schema. The agreements made by the developers of these large BIM authoring tools are named MVD (Model View Definitions). There is only one important MVD in the context of projects like CHEK and that is the Reference View MVD.

6.3 IFC MVD options

To make things more complicated there are agreements on variants of different content allowed and available within an MVD. In our case it makes a lot of sense to request BIM applications to at least generate 'second order space boundary definitions' within the exported IFC file. Modern CAD systems very well understand the object a designer is creating, it therefore normally understands the concept of a room and its relation to its binding components like walls, floors, doors and windows. Adding 'second order space boundary definitions' into the IFC file allows the BIM system to export its knowledge on the room and bounding objects as well as defining the shape of the actual boundary faces. In the case of Digital Building Permits this knowledge is useful in many situations. It is complex but not impossible to create such data if it does not exist yet in the IFC file, however having the BIM system generate this data will reduce potential errors as well as the complexity of the rule engine as created in WP4.

Concluding the final requested setting in the BIM system to create a proper IFC file is:

IFC4 ADD2 TC1, Reference View MVD + second level space boundary

Next to that it is expected that the designer is using proper classification of its objects, a wall should be defined as a wall and a floor as a floor. If the designer follows proper classification, the BIM system ensures the proper IFC classification is used on exporting the IFC file. What a proper classification is depends on the BIM system used. This

does not necessarily mean the classification of the BIM system follows the same classification as used by the IFC schema.

With the above choices the IFC file received, i.e. a computer interpretable version of the design / BIM model is created. The beauty is that we can put high trust in the content, the structure and completeness of the data available. Compared to many other open and closed standards IFC is very restrictive (a large set of agreements need to be followed on generating an IFC file and incorrect use can be recognized). This is a real pain for the people having to implement support for this standard but a large benefit for CHEK and further processing of this data.

6.4 IDS

Although the above restrictions deliver us very useful data as a base for the rules to be executed for the Digital Building Permit checks, CHEK IFC is still incomplete. Each specific rule has a specific set of input requirements. We need to know what these requirements are, how they map towards the IFC data available and for the information not available 'by default' how to create or request this data. This is exactly where the Information Delivery Specification (IDS) and micro-services play their role.

IDS allows to define 'extra' rules on top of the IFC schema / MVD that require the IFC file to contain that data that is required to calculate the rule. This way we can be sure that once the IFC file is valid against the IDS it will provide enough data to be able to complete the calculation of the rule. Note that it does not say anything about the quality of the data for a specific use-case.

6.5 Micro-Services

Micro-services can be an excellent tool to calculate and validate alphanumeric values given by the designer with geometry available. Especially in cases where implementing a micro-service as a generic solution that can be applied in all cases takes considerable effort and falls outside the scope of CHEK it makes sense to implement them in alpha stage covering the most common simple cases and use them as validation of input given by the designer. These micro-services are typically part of Task 2.5.

In some cases, the development of a micro-service that covers all cases is relatively easy, or the benefit for designers not having to bend over backwards to populate proper CHEK IFC files is so major that these micro-services are essential to be part of CHEK IFC itself. This means these micro-services automatically update / adjust / populate IFC files in a way they commit to the CHEK IFC relevant IDS file.

7 Conclusion

The definition and specification of CHEK IFC has proven to be a challenge. The moving world of open standards and strong interest in requirements modelling forced proper state-of-the-art research on current solutions concerning open standards.

With the help of CHEK's advisory board and all partners in WP2, we made a proper choice using IDS 1.0 and micro-services. The IFC standard from bSI knows many flavors and even after finding a clear winner, there are many choices to be made on MVD and options. The final choice was IFC4 ADD2 TC1, Reference View MVD + second level space boundary.

Validation of the complete IFC schema definition and the validation of IDS is essential and relevant to the core of the CHEK IFC definition. For this reason, tooling is created to properly apply these validations (including WHERE rules and FUNCTIONS); this tooling has proven essential to be able to define IDS files for CHEK. Further tooling for validation of CHEK IFC is part of Task 2.5.

Concerning the requirements themselves as coming from D2.1 we see a very good first set of classes and properties relevant for IDS and micro-services (representing CHEK IFC). We however do expect that CHEK IFC will be amended and improved during the implementation of rules in WP4 and pilot cases (WP6), in addition to real-life content coming from designers and municipalities. The current work done in Task 2.3 and delivered as this deliverable allows a very well-defined way how CHEK IFC can evolve and the tooling to support this work.

8 References

8.1 List of Figures

8.2 List of Tables

8.3 List of developed software

EXPRESS Schema Validation (ISO 10303-11:2004)

https://compiled.rdf.bg/ifcengine/ifcengine-latest.zip

IDS 0.9.6 Validation (Open Source using ISO SDAI)

https://github.com/I-Sokolov/RDFApps/tree/main/IDSChecker

IFC Viewer + Schema Validation + IDS Validation (Open Source using commercial IFC Engine)

https://github.com/I-Sokolov/RDFApps/tree/main/ifcviewer (https://rdf.bg/ifcviewer/ifcviewer.zip)

IFC CHEK (IDS and PSD files)

https://github.com/I-Sokolov/RDFApps/tree/main/IDSChecker/CHEK

8.4 List of used abbreviations

BIM	-	Building Information Model
bSI	-	buildingSMART International
DBP	-	Digital Building Permit
DoA	-	Description of the Action
EC	-	European Commission
EU	-	European Union
IDS	-	Information Delivery Specification

IFC	-	Industry Foundation Classes
GA	-	Grant Agreement
GIS	-	Geographic Information System
PSD	-	Property Set Definition
SDAI	-	Standard Data Access Interface
WP	-	Work Package