Technical Roadmap buildingSMART

Getting ready for the future

Version: 30 April 2020

buildingSMART International

Executive Summary

The digitalisation of our industry is accelerating. The demand and use of openBIM solutions and standards are increasing.

buildingSMART used to primarily focus on data exchange within the building domain. It is now pushed to widen the scope. More and more users, developers and modellers from outside the building domain want to use openBIM standards in their processes and tools. With new concepts like smart buildings, smart cities and digital twins just around the corner, there is an increased expectation for future-proof standards and solutions. This increased request for dealing with high data volume, low latency in exchange, modern frameworks for Artificial Intelligence and Machine Learning have quite a disconnect to the current file-based information silos.

The industry is moving towards more and more connectivity between domains. To facilitate this, buildingSMART needs to create scalable interoperability for data standards, tools and the underlying technologies.

The current standards and solutions operate in a limited environment. The threshold to use openBIM standards is high for people that are new to the community. This is often due to the use of underlying technologies that are not mainstream (anymore), the use of bespoke tools in the management process, and the creation of solutions that are not designed to work in modern, more generic development environments.

Creating a future technical roadmap is difficult today because the technology base of many standards and solutions is not scalable enough. The current technical roadmap is focussed on moving solutions and standards towards a generic, scalable base that lowers the threshold for users, modellers, developers and implementers from both inside and outside of the community to quickly and reliably use the standards and solutions.

The main goal of the short term is to move from bespoke solutions and technology to technologies and solutions that are scalable, widely adopted and work in a broad range of tools.

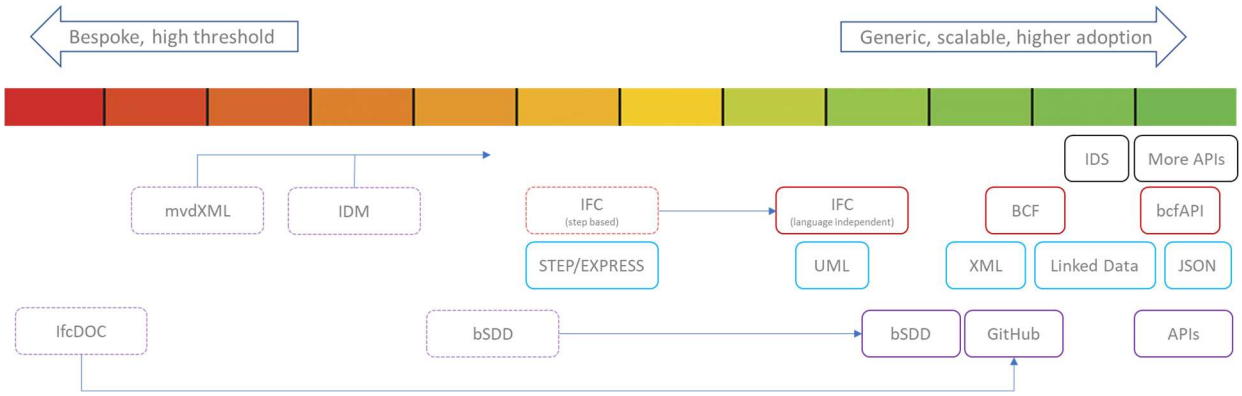
The current situation is represented in the following indicator:

This indicator shows standards in red, the underlying technology in blue, and solutions (and services) in purple. The left side represents the most bespoke standards, solutions and technologies. These have a high threshold for adoption and don't work 'out of the box' for developers and users from other domains. The right side represents the most generic standards, solutions and technologies. These have a low threshold for adoption and have the characteristic to be generic, so developers and users from other domains can work with

them. The positioning of the blue technology parts is a mix of how generic the technology is, how scalable it can be, and how broadly it is adopted. The challenge for buildingSMART is to move as many standards and solutions (red and purple) to the right side of the indicator. In many cases, the underlying technology is a limiting factor for this (the blue boxes have a fixed place).

This Technical Roadmap is focussed on transforming the current standards and solutions towards more generic technology bases. This is needed to create the necessary scalability and broad adoption.

The biggest and most important challenges are presented in the indicator below:



Separating the schema of IFC from the modelling language and file format, will force the opportunity to eliminate complex modelling heritage. From a language independent base, different serialisations to SPFF, XML, RDF, JSON and binary formats can be generated efficiently.

The roadmap also presents a solution to make the IFC schema modular to improve maintenance, create more reliable implementations and increase predictability in release cycles. It makes IFC more reliable and allows the vendors to plan implementations and decrease the time between release and availability in the tools. Modularisation has an impact on the perception of MVDs. There is a need and opportunity to create a new computer interpretable standard to define dynamic ‘Information Delivery Specifications’.

Current tools like IfcDoc and bSDD are proven to be a bottleneck in use-cases. The unfamiliar API of the bSDD can be better aligned to work ‘out of the box’ in industry wide uses like swagger. The data structure in bSDD should be more connected to IFC and facilitate the use of Product Data Templates to allow end-users to find the data faster and more reliable. The IfcDoc functionalities needed in the buildingSMART Standards deployment procedures, must be migrated to an online system. The most obvious solution would be to advance them into the Continuous Integration environment on GitHub. Online deployment and maintenance processes will provide transparency and broader engagement in the development, while forcing consistency and automated quality checking of the contributions.

There is a challenge to embrace the growth and associated changes. The technical community has always had the role of gatekeepers, guarding the independence and quality of the standards and solutions. The shift to more generic technologies to allow broader adoption should go hand in hand with the predictability of the developments and quality checking processes. Quality checking procedures and tools need to be developed so engagement from outside the current community is facilitated in a transparent and predictable way.

Contents

List of Tables

List of Figures

1 The need for a roadmap

The digitalisation of our industry is accelerating. The demand and use of openBIM solutions and standards are increasing.

buildingSMART standards used to primarily focus on file-based data exchange within the building domain. It is now pushed to widen the scope. More and more users, developers and modellers from outside the building domain want to use openBIM standards in their processes and tools. With new concepts like smart buildings, smart cities and digital twins just around the corner, there is an increased expectation for future-proof standards and solutions. This increased request for dealing with filtering high data volumes, low latency in exchange, modern frameworks for Artificial Intelligence and Machine Learning have quite a disconnect to the current file-based information silos.

The industry is moving towards more and more connectivity between domains. There is a need for 'next generation' standards and solutions. To facilitate this, buildingSMART needs to create scalable interoperability for data standards, tools and the underlying technologies.

1.1 Scope

Derived from the need and necessity for the roadmap, the scope of this roadmap is on the standards and solutions. This included the certification of software and the toolchain used for development and maintenance of the standards. It does not cover work such as the user driven IFC extensions nor the user compliance program.

The current scope of the roadmap is to set a plan for the period from 2020 till the mid of 2023. Some aspects in this roadmap highlight a longer-term perspective, to create a setting for the actions needed in the next years.

2 buildingSMART Technical Objectives

In a fragmented industry like the construction industry, the quality of data exchange is vital to improve efficiency. Exchanging data between different organisations requires reliable standards and implementations. Only with high quality data exchange the productivity of the construction industry can be improved. The quality of the exchanged data is determined by the predictability of the data schema, the semantics (the documentation and definitions of the entities) and the consistency of the implementations. All these aspects influence the eventual quality of the data that is being exchanged. These factors also influence each other. A complex schema or loosely defined semantics may result in unreliable implementations, resulting in low quality exchange.

The current standards and solutions operate in a limited environment. The threshold to use openBIM standards is high for people that are new to the community. Many people inside the community even struggle to use some standards and solutions in an efficient way.

This is often due to the use of underlying technologies that are not mainstream (anymore), the use of bespoke tools in the management process, and the creation of solutions that are not designed to work in modern, more generic development environments.

Creating a future technical roadmap is difficult today because the technology base of many standards and solutions is not scalable enough. The current technical roadmap is focussed on moving solutions and standards towards a generic, scalable base that lowers the threshold for users, modellers, developers and implementers from both inside and outside of the community to quickly and reliably use the standards and solutions.

The main objective in the next two years is to increase usability to drive broader adoption.

The strategy to do this, is to move from bespoke solutions and technology to generic technologies and solutions that are scalable, widely adopted and work in a broad range of tools.

The consequence of this objective is that some highly advanced solutions need to be deprecated to allow broader adoption and more reliable use of the solutions.

2.1 Challenges

The current situation is represented in the following indicator:

Figure 1: Current status of Standards and Solutions

This indicator shows standards in red, the underlying technology in blue, and solutions (and services) in purple. The left side represents the most bespoke standards, solutions and technologies. These have a high threshold for adoption and don't work 'out of the box' for

developers and users from other domains. The right side represents the most generic standards, solutions and technologies. These have a low threshold for adoption and have the characteristic to be generic, so developers and users from other domains can work with them. The positioning of the blue technology parts is a mix of how generic the technology is, how scalable it can be, and how broadly it is adopted. The challenge for buildingSMART is to move as many standards and solutions (red and purple) to the right side of the indicator. In many cases, the underlying technology is the limiting factor for this since the technology (blue boxes) cannot be influenced and has a fixed place. The consequence is that some of the current standards and solutions will need to be based on a more generic technology stack.

The challenge is to transform the current standards and solutions towards more generic technology bases. This is needed to create the necessary scalability and broaden adoption.

The biggest and most important challenges are presented in the indicator below:

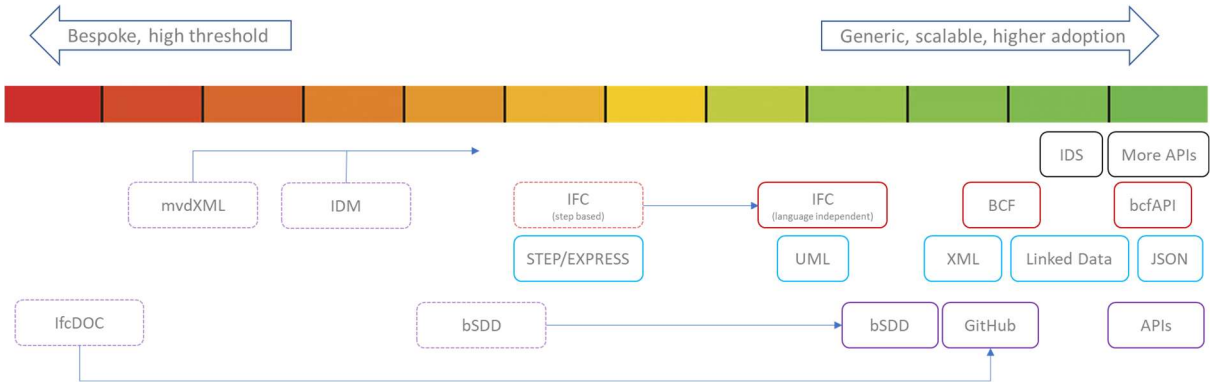


Figure 2: Desired status of Standards and Solutions

IFC needs to eliminate bespoke modelling techniques that have not equivalent in other modelling languages, to be able to become more widely adopted. Separating the definition of IFC from the modelling language and file format, will force the opportunity to eliminate complex modelling heritage. From a more generic base, different serialisations to SPFF, XML, RDF, JSON and binary formats (like HDF5) can be generated efficiently.

2.1.1 Modularisation

The constant growing schema of IFC makes it hard to maintain and deliver new releases. The release cycling of IFC is getting longer with every extension. The challenge is to make the IFC schema modular. This will improve stability and quality assurance during maintenance and create more reliable implementations and increase predictability in release cycles. It makes IFC releases more reliable and allows the vendors to plan implementations and decrease the time between release and availability in the tools.

Modularisation has an impact on the perception of MVDs. This pushes the need to rethink MVDs and the conformance level of IFC implementations in software. There is a need and opportunity to create a new computer interpretable standard for the definition of '*Information Delivery Specifications*'.

2.1.2 Services

Technical services like the IfcDoc tool and bSDD service are proven to have high thresholds for users, resulting in a very small user base.

IfcDoc is a standalone desktop tool for windows. It is custom made for buildingSMART and only a handful of people understand it. There is an increasing request to develop the

standards online, with proper traceability of decisions and version management. The functionalities of the IfcDoc tool should therefore be migrated to work in a continuous integration environment like the capabilities that the GitHub framework provides. This will provide transparency and traceability in the development while forcing consistency and automated quality assurance of the contributions to the standards.

The bSDD data structure should be more connected to IFC and facilitate the use of Product Data Templates to allow end-users to find the data in the bSDD faster and more reliable. The purpose and the objectives of bSDD should be communicated better, and the user base need to grow to keep it a sustainable service for the future.

The consequence of the objective to scale reliable solutions and standards, is that some highly advanced technical solutions need to be replaced with alternatives that are easier to adopt. This will allow broader adoption and more reliable implementation and use of the standards and tools. There is a challenge for the community to embrace these associated changes. Technological simplification, and use of lesser advanced solutions is difficult, but a necessity for a sustainable and reliable future of the solutions.

The shift to more generic technologies to allow broader adoption should go hand in hand with the predictability of the developments and quality checking processes. Quality checking procedures and tools need to be developed so engagement from outside the current community is facilitated in a transparent and predictable way.

All these challenges will be addressed in the following chapters.

3 Current Standards

3.1 Evolution of IFC

The Industry Foundation Classes are the flagship standard of buildingSMART. It is the most mature standard that has a long history and many implementations in software. IFC has always been defined using STEP technology. The STEP standard is very efficient and has a rich set of advanced modelling techniques to create efficient file-based data exchange.

3.1.1 'Next Generation' IFCs

While the current structure of IFC is focussed on creating a standard for file-based exchange, new ways of working with a connected Common Data Environment (CDE), application of Digital Twins, connections to (streams of) sensor information, micro services, and future Smart cities, are demanding new requirements to IFC.

The use of a CDE, or the use of IFC in a Digital Twin requires an object-based use of IFC data. The current IFC is optimized for file-based exchange. To facilitate current use-cases like working with connected CDEs, and new business concepts like Digital Twins and automated (micro)services, IFCs needs to become capable of transactional exchanges, allowing smaller discreet exchanges.

Transforming IFC to be capable of being used in a transactional environment is a huge task, and a drastic shift from the file-based optimization modelling techniques that are used up until now.

Transactional capable IFCs can still be exchanged as files but also accessed, maintained and exchanged using an 'Application Programming Interface' (API). Partial (transactional) file exchange, or exchange of 'changes' (sometimes called 'deltas') is an industry need already. CDEs and Digital Twins are strongly based on the connection of different systems using APIs. The standardisation of APIs is an activity buildingSMART should focus on in the coming years. Currently it is difficult to use IFCs through an API because the way it is structured. Exchanging partial files between systems is also difficult because of the complex structure of the files. The STEP specific modelling techniques used to optimise file-based exchange is hindering the object-based access and exchange of (partial) IFC.

Changing the objective to optimizing IFC to 'be used in a transactional environment', instead of 'optimizing file-based exchanges' is a big cultural change. It means the tech community of buildingSMART needs to use other key performance indicators during the development of IFC than they have been used to for the last 15 years.

3.1.2 The future of the IFC schema

Currently, the data model of IFC is very closely related to EXPRESS modelling techniques. This makes it hard to represent IFC in different formats than STEP. The focus on optimization of file-based exchange has resulted in very advanced data modelling solutions.

Using the STEP standard is the best choice to model the IFC schema when it is focused on file-based exchange. The use of very advanced methodologies that are available in STEP, provides the opportunity for small file sizes and efficient file storage.

The consequence of using these highly advanced techniques is that the schema becomes very complex. This complexity has numerous negative side-effects:

- The geometry kernel is too big to fully implement. There are many specific entities that increase the efficiency of storage, but only have a few use-cases.

- The structure has many dependencies in it. The final representation of entities is depending on the attributes of others (for example positioning of objects). The whole file needs to be analysed, with multiple dependencies, before the result of a single object can be derived.
- The use of advanced data modelling structures like '*linked lists'*, and '*selects'* have little or no comparable equivalents in broadly used languages like UML and linked data standard RDF. Trying to mimic the IFC structure in languages like XML and JSON creates bespoke solutions that cannot be used with out of the box libraries.
- The many advanced data modelling techniques that are available provide a wide range of options for software vendors to build implementations. This creates different implementations in software that are not interoperable. The schema needs to be stricter and eliminate ambiguity.
- Many of the advanced structures in IFC have been proven to be too complex for software vendors to implement. The time that vendors need to implement IFC is too long, and for vendors that have lower commercial interest in supporting IFC the threshold becomes too high. This results in half-baked implementations that cause problems for end-users.

To guarantee a future proof IFC, the schema needs to:

1 become predictable and consistent (to become stricter and easier during implementation);
2 be based on methodologies that have an equivalent in multiple, modern computer interpretable languages like XSD, OWL and JSON (see Figure 3);
3 remove circular references and possibly add identifiers to entities that don't have any now;
4 become modular to facilitate additional domains and extensions.

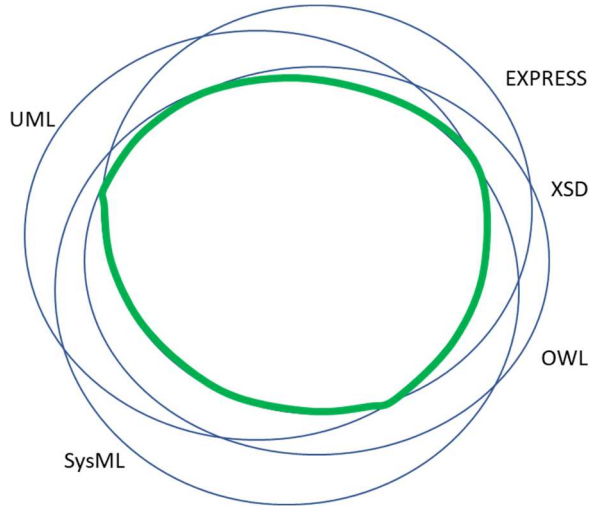


Figure 3: The overlap (bold line) between different machine-readable languages should be the base of IFC. Specific modelling techniques that only exist in one language should be avoided as much as possible.

Separating the structure of the IFC schema from the modelling language and file format, will force the opportunity to eliminate complex modelling heritage as well. From a more generic base, different serialisations to SPFF, XML, RDF, JSON(-LD) and binary formats can be generated efficiently. This will also open the possibilities for data modellers from other domains to contribute to the development of IFC. This creates a broader community that is scalable for future challenges.

Separating IFC from the underlying technology means that the STEP specific modelling techniques that optimize file sizes are not available anymore. This will impact file sizes, but it is expected this will be around 5%. File sizes in practice will actually reduce due to the ability to create partial models easier. Another option to reduce file sizes is to actively push the binary serialisation of IFC. The HDF5 serialisation is an official STEP part to exchange indexed binary IFC, but is not used much due to practical reasons. Many claim that file sizes are not a problem anymore with the rise of large (online and offline) storage abilities. Many users complain about file size, but research learns that it is actually the time it takes to generate an IFC file that is the hindering factor. Removing STEP specific modelling techniques, adding identifiers, and removing circular references in the schema, will probably improve the speed of IFC export from authoring tools.

The split between the conceptual IFC schema and the modelling techniques needs to be sensible and reasonable. It is of vital importance to involve important stakeholders like the software vendors, IFC Extension projects and other implementors. Depending on how thorough the modernisation of the schema will be, it could also provide a solution for the long-standing discussion about round tripping with IFC data.

3.1.3 From monolithic to modular schema

IFC originally focussed on the standardisation and exchange of data in the building industry. It has a '*Control extension'* (building services and component elements) and a '*Process extension*' (management and facilities), but the main elements are the '*Building Elements'* from the '*Product Extension'*. In recent years, there have been more and more extensions added to the schema. *IfcBridge*, *IfcRoad* and *IfcRail* as the leading most prominent, but *IfcTunnel*, *IfcLandscape* as well as *Ports & Harbours* are also in development.

Software implementations are not required to implement the full schema, but a subset of the schema that has additional restrictions. These so called '*Model View Definitions'* (MVDs) are currently the basis for implementation and software certification. MVDs are defining three things: (1) a subset of the full IFC schema, (2) additional restrictions to it, and (3) the conformance level that is expected of software implementations.

This makes the current approach to MVDs unsustainable. Users don't always know that an IFC Dataset is based on an MVD, and have difficulties understanding why they are not interchangeable. Multiple MVDs do not guarantee interoperability between each other. Every MVD needs to be implemented as a separate feature in software tools, which does not make it scalable. Additionally, the monolithic schema of IFC requires everyone to agree on everything before a new version of IFC can be released.

To solve these issues, the IFC schema needs to become modular. Modularisation of the schema makes it easier to separate responsibilities, and distribute the maintenance of the entities, and possibly even have separate release cycles per module. The three functionalities that are currently provided by the MVDs need to be further developed in separate, coordinated initiatives. Modularisation will facilitate this by creating a shared (interoperability) layer in the schema as a base for the modules. The specialization structure of IFC makes it possible to have dynamic ('late binding') modules that extend the base layer.

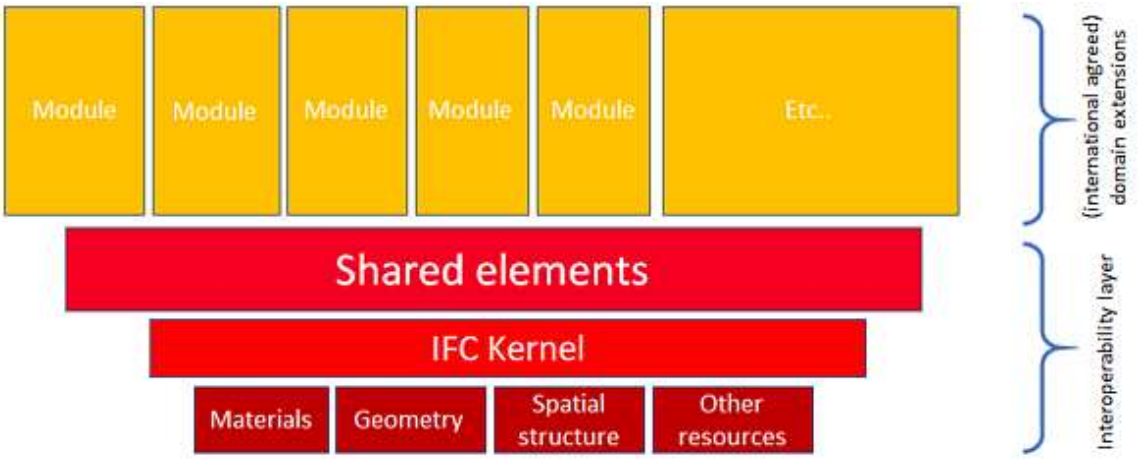


Figure 4 Modules (yellow) on a shared base (red layers) creates interoperability

This will make the implementation in software more predictable. A conceptual representation of modularisation is provided in Figure 4. It shows the Interoperability layer, comprised of the three red layers in the picture. The dark red layer represents the IFC resource layer.

When the shared base is implemented, the modules are extensions to define additional classification and properties on top of the shared layer. The split could be done on all specialisations from *IfcBuiltElement*, or possibly on *IfcProduct* when the implications are not too drastic. Other branches like the tree behind *IfcControl* and *IfcProcess* need to be reconsidered as well.

When the red interoperability layers are implemented in full, no matter what module the software supports, interoperability between domains will be guaranteed. Strict implementation of this shared base is crucial. Therefore, this part of the schema needs to be revised to create a base that is straight-forward, with minimal modelling complexity to make it stable and predictable to implement. The modifications listed in paragraph 3.1.2 need to be applied to IFC. It still needs to be investigated to determine what parts of IFC should be in this shared base, and what is realistic to ask from vendors to fully implement.

Strict implementations should also be enforced using *Implementer Agreements*. Historically, the *Implementer Support Group* (ISG) produced Implementation Agreements as additions to the IFC schema and specifications, to create stronger interoperability between implementations. The creation of implementer agreements (by vendors) should be a stronger part of the governance and certification procedures of IFC.

When the shared base is implemented in software according to a strictly defined conformance level (See 3.1.5), every export from any software will provide a dataset that can be imported in any other software tool with the same conformance level. In other words, a '*one star*' export from '*module A'* can always be imported in any other software tool that implemented '*one star'* IFC, no matter what module is supported or certified. This creates the predictability for IFC to be the base for Digital Twins, and the ability to support use cases where data is exchanged through APIs instead of files. More on this in chapter 3.1.5.

Depending on how software vendors implement the modules (extensions with classifications and properties), new modules or updates could potentially be supported instantly. Even when this is not immediate, the time between the publication of a new version of an IFC module (i.e. extension), and the availability in software would decrease drastically.

In conclusion, advantages of modularisation are:

- Shorter release cycles of IFC;
- Faster support of new IFC versions in software;
- (instant) support of new modules (extensions);
- Stronger interoperability between modules (extensions/domains).

3.1.4 Backward compatibility

Backward compatibility has been defined as the ability to open IFC files structured in older versions, in software tools that implemented a newer version of the standard. So, a software tool that has import capabilities for IFC4 should also be able to import an IFC2x3 file using the same codebase for the IFC4 importer.

This means that every new version of IFC should only extend the current version. This is also why the 'x' that stands for 'extension' is in the name of the versions. IFC2x3 is the 3rd extension of IFC version 2. This backward compatibility has never been realized in practise. In every new version IFC entities have been removed or drastically reorganized resulting in significant effort for new implementations.

Obviously, cleaning up the schema, making it easier to implement, and transforming it from file-based optimized structure to a transaction capable structure will create changes that are not backward compatible either. The resulting schema that is more consistent, cleaner and modular, does provide a basis to better facilitate backward compatibility in the future.

To support transformation from IFC2x3 and IFC4.x to the newest IFC, every new release of IFC should have a 'transition path' documented how to transform a previous version of IFC data to the newer version. This transition path can be documented in text or, preferably, in executable reference implementations (like a script).

3.1.5 Standardised conformance levels

Different use-cases may require different levels of conformance of the IFC representation and implementations. Software tools designed to visualise IFC only need geometry definitions that are close to graphics pipeline and hardware. Structural analyses software might want to have very precise geometry constructs. Other software tools and use-cases might want geometry definitions with Boolean operations to trace the way the final representation is constructed.

For the implementation of the base layer three different conformance levels for software implementation can be defined:

Table 1: Standardized conformance levels for implementation

Level	Use case	Types of entities
One Star	Fast, reliable visualisation of data, geometry, relations, classifications and properties. Datasets can also be without geometry.	Geometry close to the functions of the common computer graphics pipelines and hardware. Probably triangles/faces + swept solids + curves.
Two Stars	One star plus the option to also define very precise analytical geometry.	One star plus non-planar surfaces, advanced boundary representation, etc. Might include NURBS, but those might also be in three stars level.
Three Stars	Two stars plus the option to define modifiable geometry,	Two stars plus constructive solid geometry (and NURBS).

	keeping the history of how geometry was constructed.	

The use of one-star geometry close to the graphics hardware enables the integration of IFC with other domains. Integrating this level of IFC geometry makes the integration with geospatial standard GML and widely used glTF geometry more feasible. Strong partnerships between buildingSMART and (for example) the open geospatial consortium (OGC) will help drive the integration between IFC and CityGML.

Optionally there might be another layer of compliance for product libraries. This might have procedural geometry with parameterizations, features and constraints allowing for a formula to calculate the parameter values and for geometric constraints (parallel, perpendicular, etc.) to be kept. This is currently not part of IFC and is not a priority for the 2 to 3-year scope of this roadmap.

The standardised conformance levels for implementation of the shared base would be natural certification levels. It is desirable/necessary to have certification per module (extension/domain) to provide a guarantee to users that the mapping of the internal vendor data model to IFC (and the other way around) is done correctly.

3.1.6 Data formats

The most dominant use-case to exchange and represent IFC is in a (container-based) dataset (a file). Although it is expected that additional means of exchange will broaden the use of IFC, files will always stay.

The proposals in this roadmap make it easier and more reliable to represent IFC data in several different file formats like XML, RDF and JSON (and even binary formats like HDF5). It is however expected that SPFF will stay the predominant way to exchange IFC datasets.

For archiving reasons, it is crucial that a text based (ASCII encoded) format will stay a priority within buildingSMART.

There might be options to persuade software vendors to always have a SPFF option for export and import when they implement IFC in the compliance program (certification). This needs to be investigated further.

3.1.7 Certification and release cycles

The modularisation and the standardisation of the conformance levels leads to a clear proposition to the software vendors and creates a scalable maintenance structure. This will lead to shorter release cycles for the modules (extensions) that are easy to process for implementors, and stability on the underlying shared layer.

The changes mentioned in 3.1.2 will create a consistent, predictable and cleaner IFC schema in the shared layer and recourses. This will decrease the complexity during implementation and decrease the complexity for certification. The creation of implementer agreements (by vendors) should be a stronger part of the governance and certification procedures of IFC.

Certification needs to be done on the standardized conformance levels. It must be defined in more detail if software tools will be certified per module, or only on the shared layer.

Lowering the threshold for certification is important to get more tools engaged. Certification on the one-star conformance level (without additional modules) could be executed by online

tools, or community members. This might bring in more software tools to apply for this level of certification.

This will result in an optimized process of IFC changes, releases, implementations, certification and usage by end-users.

3.1.8 Deployment strategy

The proposed changes to IFC will have a big impact. Even the smallest edits to the schema will create a ripple effect through the whole ecosystem of extensions, certification and implementation.

There are multiple ways to deploy the new strategy. There are two main ways to do this:

Table 2: Two main deployment options for the new IFC Strategy

Step by step	Integrated
This means creating a simple MVD for the new version of IFC, without actually removing or changing entities in the Schema. This simple 'test version' can be used to see the impact of the changes. Only after verification the next step will be taken. When the new version is tagged as final the changes in the certification could start. This step by step procedure will take years before the desired changes will be in effect and end-users will notice them.	Because the proposed changes are all very connected, it could be argued that one big step is a way to deploy them. Making the IFC Schema consistent and modular will make it easier to separate the conceptual model and the formats; will create easier implementations and certification. Depending on how thorough the clean-up of the schema will be, the cascading effects to maintenance, implementation and certification are speeding up the deployment. This option should only be considered when there are enough recourses and a strong collaboration with the vendors and Implementor Support Group.

When the next generation IFC is more predictable and has a lower threshold for implementation, it is to be expected that software vendors will wait for the version that requires the least amount of resources to implement. The step by step approach might take longer when vendors don't act on the intermediate steps. The integrated approach is only possible with enough resources and funding.

The roadmap is not able to suggest the best deployment strategy. This needs to be decided by the buildingSMART stakeholders.

3.2 Evolution of BCF

The BIM Collaboration Format (BCF) allows different BIM applications to communicate model-based 'issues' (formally called 'topics') with each other by leveraging IFC data that has been previously shared among project collaborators.

There are two different ways to utilize BCF – via a file-based exchange or via a web service. The file-based exchange workflow is relatively straightforward and is a process most people are used to using. A zip file is transferred from user to user, edited and returned. As an alternative to the file-based workflow, there is the web service-based (RESTful) API mode for BCF. This involves the implementation of a BCF API endpoint.

In the formal schema of BCF, there are a couple of mandatory fields, but many fields are optional. This means vendors are free to export many of the fields, but do not have to. The same applies to the import; the mandatory fields need to be supported, but the optional fields are not always imported into the software tool. This causes a situation where the exchange

of a rich BCF dataset can lose data during import and export. The de-facto situation is that round tripping is not possible without loss of data. The BCF standard has had no significant updates in recent years.

To facilitate the continuous development of BCF it should be better integrated in the buildingSMART governance process. A formal group needs to be established that drives the development and releases of BCF. This group should be a mix of software vendors and end-users to guarantee user-driven features and developments. A formal BCF certification can be facilitated by buildingSMART, after the governance structure is established and mature.

3.3 Evolution of other standards

The described changes in IFC will drive the ability to create object-based data exchange. To further facilitate this, some other elements are needed:

- An IFC query language;
- Standardized APIs (to allow moving towards BIM level 3 / integrated deliveries);
- Workflow standards

A query language is only possible when the changes in IFC are processed. There is a strong relation between an IFC Query language and object-based API development. For an object-based API there needs to be a stable foundation and established file-based API first. Therefore, the query language development is probably out of scope for the next 2 to 3 years but mentioned as the objective for the development of the overall API strategy, and the transformation of IFC.

3.3.1 APIs

Many of the current buildingSMART standards are focused on file-based exchange. Current technology with the use of connected data environments and upcoming requirements for Digital Twins, Smart Buildings and Smart Cities, require a more 'data driven' approach.

To facilitate the interoperable exchange of data standards (like IFC, BCF, bSDD data, and dynamic Information Delivery Specifications) an overall API strategy needs to be in place.

The BCF API and the openCDE API have identified the need for a shared, common API. Both are strategic projects to work towards an overall strategy for buildingSMART APIs.

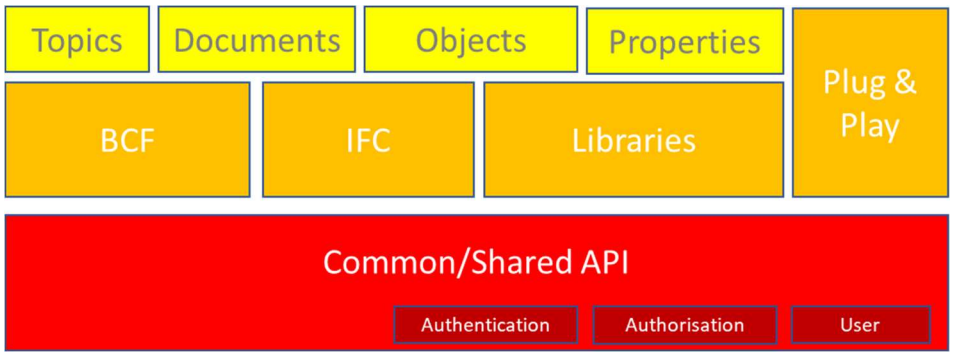


Figure 5: The overall API Strategy

The current versions are still focussed on exchanging files through an API endpoint. In the future a more object-oriented API need to be developed as well.

Besides these components that are in development, a standardised API for libraries should be developed. bSDD should support this API as well.

Increased use of APIs for exchange of datasets, leads to an increased interest in cyber security solutions. BuildingSMART will not develop separate security standards for data exchange, but adopt available industry standards for digital signing, encryption, authorisation and authentication.

3.3.2 Workflow standards

There are many workflow and process standards for managing information flows in the built environment. The standard for processes in buildingSMART is the ISO 29481 on the Information Delivery Manual (IDM).

Other standards are the ISO 19650 series, the ISO 23386 to describe, author and maintain properties in data dictionaries, the upcoming ISO 23387 to define data templates, the upcoming CEN 17412 to define Level of Information Need, and many others.

All of these standards have their own purpose and methodology. Some focus more on the process maps, some on the transactions or a methodology to define something.

In the industry today, many BIM users define their own BIM Execution plans, including process maps with exchange requirements. While the process flows are often defined using a computer interpretable Business Process Modelling Notation (BPMN), the definition of exchange requirements is often only human readable.

There is no computer interpretable standard to define a '*Information Delivery Specification*'[1] (IDS). In the period of this Technology Roadmap, buildingSMART should be focussing on developing a machine-readable standard to define '*Information Delivery Specifications*'. Potentially, such an IDS standard could be used in most of the above-mentioned workflow and process standards.

3.4 New standard: Information Delivery Specification

Exchange (Information) requirements can be dynamic per use-case and per project(phase). Currently, there is no suitable (computer interpretable) standard to define such '*Information Delivery Specifications*' (IDS).

Historically, there are two options to define data requirements, essentially filtering a larger schema into a smaller set. Within IFC this is mvdXML, and within the bSDD it is done using '*contexts*'.

The mvdXML definition was intended to configure IFC exports from Authoring tools that support full IFC. It was not developed to define data requirements from end-users, although some features to facilitate that have been added later in the development. The mvdXML definition is proven to be very bespoke and not suitable for practical exchange requirement definition by end-users. Some known issues are:

- It lacks some important information, such as data types for properties.
- mvdXML assumes that the software used for filtering and validating has IFC as the internal data structure, but this is almost never the situation. The format for the definition of exchange requirements should not be 1:1 tied to IFC schema, but to the concepts used in IFC.
- Real requirements are mostly not set for object classes, but for groups of objects. For example, external walls must have a U-Value. To do this the first requirement is that it must be possible to find the external walls from the model. External walls may be

[1] See the definition of 'Information Delivery Specification (IDS) in Annex A: Terminology

walls or curtain walls, or of any object class. With the external walls, the requirement for the U-Value (value for thermal resistance) can be set. This may also be cascading, for example certain physical objects must have a material, if the material is concrete they must have concrete properties and if the object is outside the building it must have additional concrete properties, like frost resistance, that are not needed for concrete objects inside the building.

- Some of the constraints are not defined using XML standards, but proprietary bespoke grammar and even string-based grammar.
- It has deep nesting and cross-referencing in non-standard ways.

The current known implementations of mvdXML cannot read each other's output because the definition and documentation are not strict or clear enough. While mvdXML is intended to be used in the backend of software implementations, the '*Information Delivery Specifications*' standard needs to operate on the front-end, where end-users can configure the output of an authoring tool. It seems difficult to use the current mvdXML as the base to define a standard for '*Information Delivery Specifications*'.

In the bSDD the '*contexts'* are being used as filters. These *contexts* are owned by specific owners and are difficult to maintain. The objective was always to standardize the context views of bSDD, which also makes them not suitable to use as base for a dynamic '*Information Delivery Specifications*' solution.

buildingSMART should develop technology to create the ability to define machine-readable requirements based on IFC, with additional classifications and properties from bSDD, and even project- and user-specific properties from outside bSDD and IFC. The technology should facilitate the definition of additional business requirements.

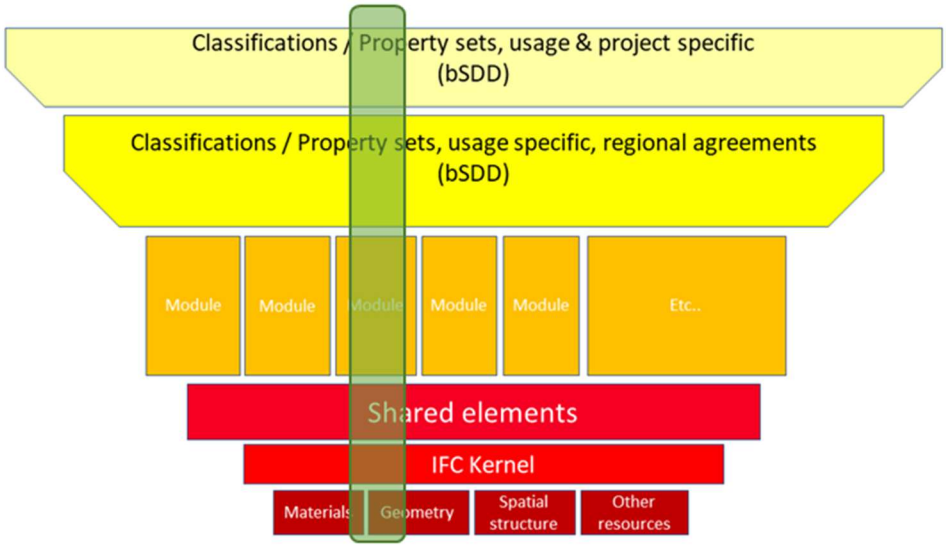


Figure 6: Information Delivery Specifications should be the filtering IFC, extensions (modules), bSDD data, and self-created properties and classifications.

Some of the current MVD projects can be expressed as standardized '*Information Delivery Specifications*', since they are often a collection of IFC entities, properties and classifications without further restrictions or alternative conformance levels. Expressing these projects as IDSs allows flexible adaptations for local markets and avoids the need to ask vendors to go through lengthy and costly procedures of custom implementing MVDs.

There can of course still be standard defined exchange requirements for use-cases like facility management and fabrication. The COBie standard is an example of a standardized

IDS with the specific purpose of building operations and facility management. Using a machine-readable IDS to express international standards like COBie would make them available in software that implemented those parts of IFC.

Several use-cases all provide input for the development of an IDS standard. Such a machine-readable standard to define and exchange IDS data should:

- Be able to define instance level requirements;
- Be able to link to bSDD concepts, properties and domains;
- Be able to specify properties and specific classifications;
- Should be machine readable to be able to load into authoring tools to facilitate both users and software tools to generate, validate and correct mapping of internal data to the desired output;
- Should be computer interpretable to allow automatic validation of IFC against the requirements;
- Should be based on industry standard technologies to work with generic parsers;
- Should be extendable;

There have already been some prototype developments with defining IDSs using generic technologies like JSON Schema, SPARQL, SHACL, Schematron and XSLT. First implementations have already been done on JSON Schema. These initiatives, just like the lessons learned from mvdXML and bSDD Contexts, together with the use-case management tool and the IDM Toolkit, can provide valuable input to this new development.

4 Technical services

The Technical Services group under buildingSMART International hold the Model Support Group, the Implementer Support group, the certification program, deployment tools and the Data Dictionary.

4.1 Standard maintenance & Certification

Under the technical services the Model Support Group (MSG) is responsible for the continues development and maintenance of IFC. The Implementation Support Group (ISG) is a group of vendors that support each other and share experiences in the implementation of IFC. Until some years ago the ISG produced Implementation Agreements to improve interoperability in implementations for cases where the IFC schema was ambiguous.

Both groups should have a strong relation. Historically these two groups have separated. The feedback and experience from vendors is and should be a strong driver for the further development of IFC. Vendors are important stakeholders that are allowed to work on the development of IFC.

Besides the meetings with the ISG, development of IFC should be on a public website, with transparent discussions and decision making. The use of GitHub for these kinds of processes is very common, so buildingSMART will start using GitHub issues and pull-requests for this. The GitHub platform will provide more transparency and accountability for the community.

The certification team also has a strong knowledge position about practical issues in IFC. They see even more detailed differences between implementations and can provide input to changes in IFC based on that.

The ISG and MSG should work very strong together. There is no historical reason anymore not to collaborate more intense. The ISG meetings can be rebranded to ‘IFC Development meetings’ with actual IFC change proposals on the agenda for discussion and voting.

The certification team usually come later in the process but should also be involved during these meetings.

The governance structure and processes for IFC are well established. For other standards like BCF there is not such a strong structure in place. To facilitate the continues development of BCF it should be better integrated in the buildingSMART governance process. A formal group needs to be established that drives the development and releases of BCF. This group should be a mix of software vendors and end-users to guarantee user-driven features and developments. A formal BCF certification can in place with buildingSMART, after the governance structure is established and mature.

4.2 buildingSMART Data Dictionary

In interoperable data exchange there is always a balance between the amount of consensus that can be reached, the adoption of the standard, and how many specific use-cases can be supported.

An international standard that needs large consensus is usually relatively small and very generic. This increases a broad adoption. The geometry standard used in IFC for example is very generic and used in many other industries. On the other side, the one-to-one agreement between just a small number of people can be very effective for specific use-cases that only exist in a niche environment. An agreement on what property to use for the

exchange between just 3 organisations for example can be created very fast and can be very useful but is only effective for just the 3 organisations.

The more generic and broadly used an agreement is, the more it becomes a *standard*. The more specific it is, the more it stays an *agreement* or *specification*.

Somewhere in the middle are the national classification systems (Coclass, NlSfb, OmniClass, etc), the agreements within specific industries (ETIM, etc). These can be called *standards* within the domain for which they are used.

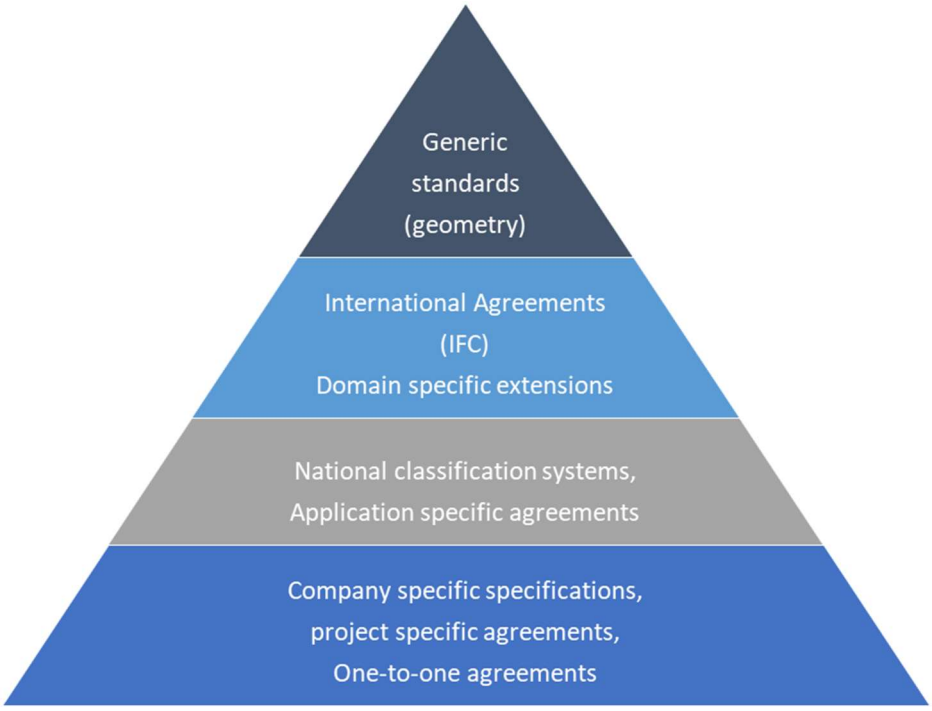


Figure 7: The balance between generic and international standards (top) to use-case specific or regional agreements (bottom)

International standards often have implementations in software. The internal structure of the software is mapped to the standard to facilitate efficient and reliable usage without manual mapping. IFC is implemented in many of the BIM software tools. It is impossible for more software vendors to directly implement all the regional or domain specific standards and agreements. Usually the classification is done by the user by 'tagging' objects according to the specification they need to work with.

Definition and maintenance of these standards can be local. To maintain the interoperability with the 'foundation classes' it is important to have a relation to the representing IFC entity.

The definition of a '*wooden exterior front door'* is different per country, and the properties attached to it might be different depending on the use-case. It is of great value, to define that this special kind of door is still a special representation of an IfcDoor. In some cases, users may have the need to define project-specific, or company-specific set of specializations and property sets.

To facilitate the connections between these classification systems and property set definitions, the bSDD is created.

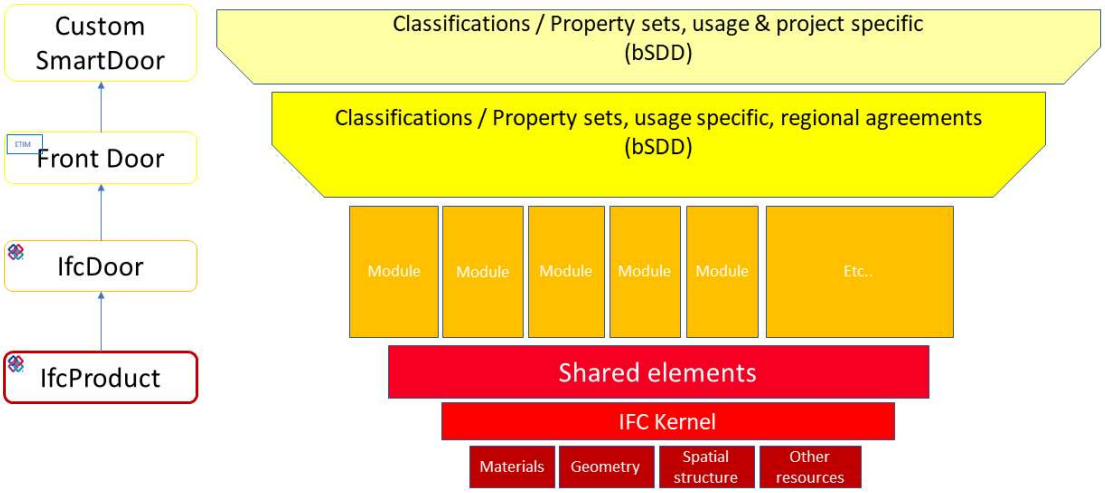


Figure 8: The specialisation tree, with the role of the IFC resources, IFC interoperability layer (red), IFC Extensions (dark yellow) and the bSDD domains (yellow)

bSDD can hold data that extends the specialization tree of the Industry Foundation Classes. So bSDD classifications can be extensions of the Industry Foundation Classes. It is also possible for data to exist in bSDD without a link to IFC.

The pyramid in Figure 8 is the same as the rotated pyramid of Figure 7. Both have the range from generic standards to international and national agreements. This specialisation and inheritance structure are the key to effective interoperable data exchange.

4.2.1 Role of the bSDD

The buildingSMART Data Dictionary (bSDD) is created to help users bridge the gap between the more generic standard that is implemented in software, to the more specific agreements that they use to enrich the semantics of a dataset.

Users can use the bSDD as a single point of entry to find the correct classifications, or properties to add to objects. The bSDD standardises the workflow of adding additional semantics to the BIM dataset. Users always have the same options to search and filter through multiple regional and domain specific standards. They can find and use the additional semantic enrichment they need for their specific use-case in the same way for every project.

Because most of the entries in bSDD have multiple translations, users will be able to find the correct data by using their own natural language during a search. When data inside the bSDD has a relation to more generic standards (like IFC), the bSDD can even automatically suggest classifications and properties to improve the semantic enrichment process even more.

This drastically increases the productivity of the users that need to enrich BIM datasets.

4.2.2 Future of bSDD

In the future, multiple use-cases that increase the effectiveness of the industry can be envisioned using bSDD. Automatic compliance checking can be done when using the right Product Data Templates (PDTs) and classifications from bSDD.

Current use-case based requests like automatic translations of objects to other languages is a feature that could be realized in the near future. This will increase the ability for people in different countries to work together without translation issues.

bSDD is mainly used in the phase between generic design/engineering and the selection of products to procure. This phase of additional specification is ideal to link the data to product manufacturers. A possible link to GS1 might help the industry with (semi-automatic) product selection.

In the future there might be a federated infrastructure of interconnected libraries with standardized APIs. When data inside the bSDD is linked to more generic standards, and linked to each other, there might be an option for automatic transformation of data to another classification system.

When IFC is used to describe data in product libraries (see 3.1.5), the properties of concepts in bSDD might be used to describe physics and geometry of IFC products. This will increase the use-case applications of bSDD.

4.2.3 bSDD Challenge

The current status of the bSDD is not operating at a preferred level. Although much of the structures are in place, it is not facilitating the use-cases.

The first step is to get a stable bSDD up and running for end-users. The underlying database needs to be restructured to facilitate the publication to Linked Data formats, the support of the upcoming CEN standard on Product Data Templates, and the actual implementation of the governance model. In a later phase additional features can be added.

The development of a 'next generation bSDD' is split up in two phases:

1) Getting a working platform which is stable for current use-cases and with effective governance in operation. This phase will result in a stable base for further development.
2) Expanding the features, creating a distributed infrastructure and connecting to other systems like GS1.

The main priority is to deliver a reliable and extendable bSDD.

4.2.4 bSDD Governance

The bSDD hold data from multiple owners. These owners are all responsible to keep their data in bSDD up-to-date and stable. The data needs to adhere to strict quality guidelines and go through a publication process with quality checks. In the database there should be a strict split between the data of different owners to make sure only owners (and their representatives) can edit their own data.

To help the data owners to achieve this, buildingSMART provides the option to work with '*bSDD Agents'*. These agents are trained to work according to the high-quality standards, and actively invest in continues development of the bSDD. This will help to guarantee the reliability of the content inside the bSDD.

The operating model is illustrated below.

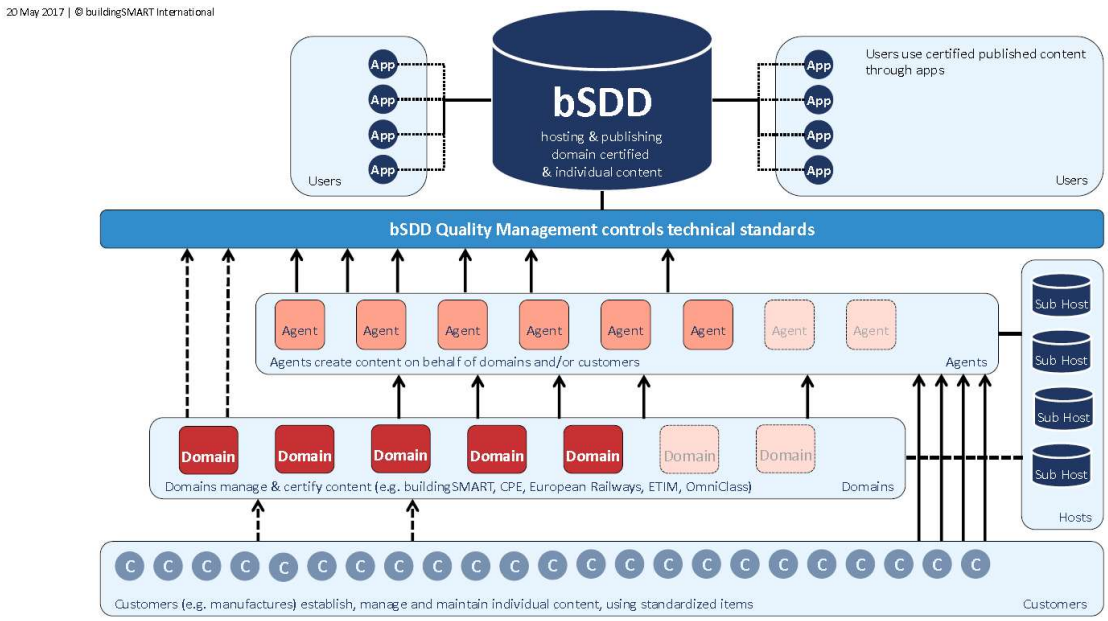


Figure 9: Operating model of the bSDD

As this model clearly states, the data from different owners, should be stored in different domains. This allows for a clear distinction between similar named concepts with different semantic descriptions. The different translations will also be strictly linked to the original semantic concepts.

To operate this model, a team of people is needed to support the agents, perform additional quality checking, administration and monitoring.

4.2.5 bSDD redevelopment

The 'next generation' bSDD needs the following investments in redevelopment:

Table 3: Development phases of a 'next generation' bSDD

First phase development	New database; Restructuring existing data; New tooling; Enable primary use cases; Licensing and business model; Quality management and governance model
Second phase development	Plugins for BIM tools; GS1 link; Automatic quality checking tools; Advanced use-cases; Business Model development and implementation

The first phase could be developed in 1.5 to 2 years after the start. A first version that can be used in practise for the first use-cases can be online after one year.
The second phase will take approximately a year and can only start after the completion of the first phase.

While operating as a buildingSMART International Service in the medium term the bSDD needs to be financially secure therefore requires a sustainable business model. buildingSMART International intends to prove a working tool based on community

investment in phases 1 and 2 above before finalising a self-sustaining medium-term business model.

A separate ‘bSDD Action Plan’ document will be created with a more detailed description of the necessary work on bSDD.

4.3 Universal Types

UniversalTypes is the name of a set of classifications and properties focussed on bringing product buyers and product manufacturers together.

The universal types dataset was originally developed by ProMaterial but is now owned by buildingSMART International.

The bSDD is made for datasets like UniversalTypes. buildingSMART has the intention to publish UniversalTypes through the bSDD. Having this dataset allows the development of the bSDD to work with a practical and use-case driven approach.

5 Governance

5.1 Deployment toolchain

The development of IFC is currently based on the use of the IfcDoc tool. This tool is custom made and grown based on the active project.

There are many pieces of the puzzle when it comes to the release of IFC. The core of IFC is developed by the *Model Support Group* (MSG), the extensions by the projects, standardized property sets (PSets) are in a separate document, translations are being made, and documentation is written by multiple people. On every release all these inputs must come together to be combined and packaged. Currently a desktop application called IfcDoc is the tool to do this. Due to the complexity and unstable nature of IfcDoc, only a few number of people are able to use it. These limitations form a bottleneck in the deployment process of IFCs.

A professional governance and deployment process demands provenance, traceability of changes and reliable version management. Given the nature of the buildingSMART community, transparency in the decision-making process is also a strong requirement.

The delivery process of IFCs should not be a one-time event, but a continues process. Putting all the developments online, provides the ability to connect them. The IFC Schema, extensions, documentation and PSets should be online. This could be done using the GitHub framework. This allows everyone to suggest changes, have open discussions and contributions. It provides the needed traceability and transparency based on strong version management and strict authorisation levels.

GitHub also provides the option to use '*Continues Integration'* (CI) tools. Every change in the schema, property definitions or documentation can trigger an automated script to run. This can perform quality checking, but also generate new content. This way the scripts can be used to generate a list of updates between releases, generate pictures and diagrams of the schema, generate or update libraries for software tools, or even generate technical documentation. Quality checking CI tools can check for the existence of documentation for every IFC entity, modelling consistency and other things. These CI tools can also be used to update the IFC entities and properties in IFC.

This will increase the transparency and traceability of developments. It also creates a more efficient and predictable release procedure. The automatic quality checks that run in the background provide more reliability and consistency in the development.

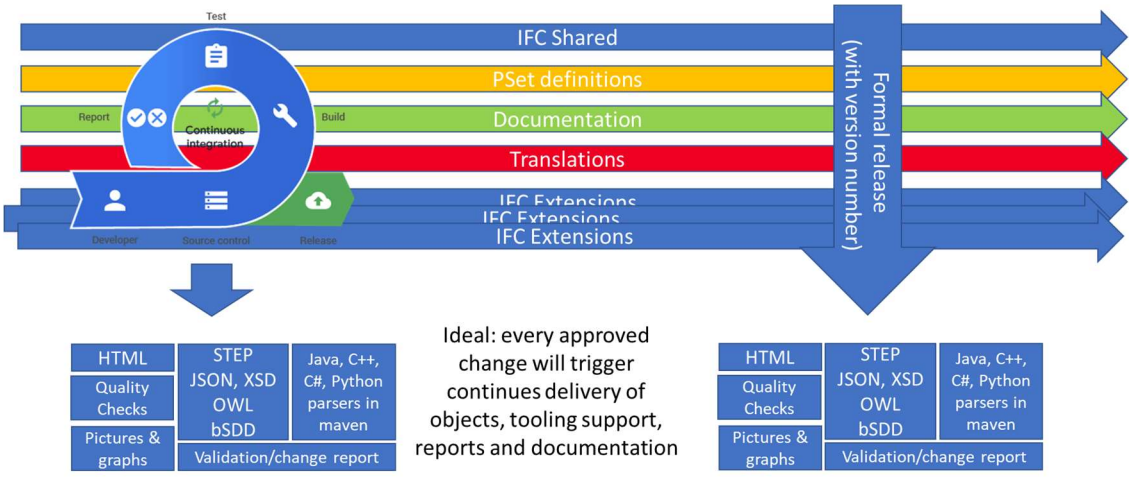


Figure 10: Continues Integration tools for the IFC deployment process

The development of BCF is already being done on GitHub. The BCF repository can also benefit from continues integration tools and automatic validations. Embedding BCF better in the buildingSMART Governance process should go hand in hand with the use of these deployment tools.

5.2 Translations

Translations of IFC have a different nature and do not reside on GitHub in an efficient way. IFC translations and quality checking should be done on a local level. Maybe the chapters can coordinate this. There are many translation frameworks that facilitate strong governance processes of machine supported translations, human fine-tuning, reviews and releases. Doing this in an online framework will keep translations in line with development and allow software tools that implement IFC to connect to the latest known translations.

Potentially there could be a business model for the chapters to host translations in their region, but this must be investigated further.

5.3 Licencing

The buildingSMART standards, solutions and services need a consistent licencing structure. In principle buildingSMART standards and solutions should be available for the whole industry, so also outside the community.

5.3.1 Standards

Standards need to be published in a way where the license does not limit the use of the standard but keeps the ability for strong governance.

5.3.2 Tools

The software tools needed in the deployment process of IFC also has different requirements. Licenses of those tools should be liberal to allow commercial use but avoid backward engineering of IFC when published under an alternative license.

Automatic generation of documentation and quality checking adds to the complexity of interlinked ownership. External legal advice should be considered for the definition of suitable licenses for the standards and accompanying toolchain.

To be able to publish standards and solutions consistently, the ownership of the content should be shared with buildingSMART. Transparent development of new standards and solution will require contributors to handover or share ownership of their contributions with buildingSMART. This process step should be an integral part of the governance process.

5.3.3 Services

The data in the bSDD are owned by different stakeholders. The publication of that data through the bSDD does not affect the ownership. bSDD is a hosting provider for the data owners. This means responsibility and liability for the use of the bSDD data should stay with the data owners. This needs to be clear for all stakeholders.

Some data owners request bSDD to make a distinction between freely available data and paid data and want the bSDD to track the usage of data. A separate bSDD action plan has put this feature request on the longer-term development. During the development of a new bSDD, a value proposition and business model need to be developed to answer licensing questions about these requests.

5.4 Quality assurance

There is a challenge to embrace the growth and associated challenges. The technical community has always had the role of gatekeepers, guarding the independence and quality of the standards and solutions. The shift to more generic technologies to allow broader adoption should go hand in hand with the predictability of the developments and quality checking processes. Quality checking procedures and tools need to be developed so engagement from outside the current community is facilitated in a transparent and predictable way.

6 Conclusion

This roadmap outlines the high-level changes needed to create a stable technology base and governance structure for the buildingSMART standards and solutions program.

In general, the challenge is to move towards a scalable technology base that can be used by a broad range of stakeholders.

The community has mainly been discussing the future of Industry Foundation Classes (IFCs). This discussion has been focused on the comparison of the underlying STEP technology with more modern technologies. This roadmap concludes that it is not about the use of STEP, or any other technology. The description of IFC needs to become independent from the underlying technology. The main challenge is to focus on the use of common modelling techniques to makes sure IFCs can be represented in commonly used languages and file formats. This objective to create a consistent and predictable IFC schema that can be used in a broad range of use-cases, is different from the objective to create efficient file-based exchanges. Obviously, this does not replace file-bases exchanges, but adds additional capabilities for the use of IFC. Bespoke modelling techniques that only reside in STEP, or OWL, or UML, or any modelling language should be avoided. This will broaden the use of IFC while keeping the current functionalities and file formats.

This way the IFC schema is technology agnostic and easier to adopt by all stakeholders in a broader range of use-cases. This will create a ripple effect in the ecosystem, also making it easier to develop and release IFC extensions and accelerate the implementations and certification process.

This roadmap also presents an integrated approach for the core of IFC, the modules (extensions) and the data in the buildingSMART Data Dictionary (bSDD). The bSDD service will help people to use open data exchange with regional and project specific requirements. The development of a computer interpretable '*Information Delivery Specifications*' (IDS) standard is a high priority to create a full circle technology stack to facilitate effective and efficient workflows.

Other standards to access the data (file formats and APIs) and communicate about the data (for example BIM Collaboration Format, known as BCF) are on a steady development track. A strong governance process will accelerate these developments. Guarding the stability of shared and common technologies like a common API and a query language will be challenging.

The proposed improvements in this roadmap require appropriate funding. A separate document needs to be generated to research the best funding strategies.

Annex A: Terminology

Preferred term	Abbreviation	Deprecated Term(s)	Rough Definition
Industry Foundation Classes	IFC	-	*Standard (agreement)* for the semantic definition of *objects*, and structuring of *information* within the *built environment.* IFC consists of the definition of objects, specialisations, decomposition, relations, properties, the definition of several file formats to exchange the data and additional documentation and specifications.
IFC Data Model	-	-	*Definition* how to structure IFC data (the core elements, resources and relations in IFC)
IFC Dataset		IFC File (some call this 'IFC Model')	*Dataset* with data structured according to the IFC standard. Usually exchanged in a file, but could also be exchanged through an API.
IFC Schema	-	-	*Definition* of the IFC Data model in a computer interpretable definition (for example, EXP, XSD, JSON-Schema, etc)
IFC Specification	-	-	*Definition* of additional information that is needed to implement IFC. This includes implementer agreements, buildingSMART defined Property Sets and extensions.
Model View Definition	MVD	-	An MVD has three purposes: (1) defining a subset of *IFC* required to fulfil a purpose (comparable with IDS); (2) adding additional restriction to that subset; (3) defining an expected level of implementation in software (comparable with Conformance Levels).
Information Delivery Manual	IDM	-	ISO 29481
Information Delivery Specification	IDS	Functional Parts	A machine-readable dataset that defines information requirements and the way they should be exchanged. This is often the combination of IFC plus additional classifications and property sets. Definitions from the bSDD can be included as well.
Conformance Level	-	-	The level of IFC that a software implementation supports.

buildingSMART Property Set	bSI_	PSet_	International agreed Property Set definition for a specific object(type) or purpose.
Property Set	-	-	Dynamic, often per project defined, set of properties for a specific object(type).
buildingSMART Data Dictionary	bSDD	-	*data dictionary* with classifications, properties, relations and translations. The data in the bSDD can be connected to IFC and to each other. The bSDD is a service of buildingSMART
Business Process Modelling Notation	BPMN		OMG standard for specifying business processes in a business process model.
International Framework for Dictionaries	IFD		ISO standard (12006 part 3) defining a framework for object-oriented information. Is was an inspiration to build the first bSDD.
BIM Collaboration Format	BCF		Standard to exchange information about the contents inside an IFC Dataset. Can be exchanged in a ZIP file (xml) or through an API (JSON).
BCF Dataset		BCF File	*Dataset* with data structured according to the BCF standard. Can be exchanged in a (zip)file, or as JSON through an API.
Open Common Data Environment Application Programming Interface	openCDE API		Standard API to exchange information between online connected data environments.
BIM Collaboration Format Application Programming Interface	BCF API		Standard API to exchange BCF data between online applications.
Industry Foundation Classes Query Language	IfcQL		Standard query language to filter IFC Datasets. Basis for future buildingSMART APIs
Use Case Management Tool	UCM Tool		Service from buildingSMART to share and find Information Delivery Specifications (IDS) and workflows (BPMN). Use-cases are organized with meta information to search and filter.
Serialisations:			
ifcOWL			OWL Ontology of IFC
ifcXML			XML formatted IFC dataset
ifcJSON			JSON formatted IFC dataset
ifcSPFF	SPFF		STEP formatted IFC dataset. SPFF = STEP Physical File Format.

Annex B: Impact analysis of IFC to current buildingSMART Initiatives

The table lists the proposed improvements to the IFC ecosystem, the priority, short description of the benefit and the impact on software implementation.

The table is mainly focussed on extension projects of IFC. MVD projects that are not extending IFC, but in fact creating a 'Information Delivery Specification' are advised to take note of chapter 3.4.

The 'Quick Wins' column shows the potential benefits current buildingSMART extension projects could get on the short term while the full roadmap is being processed. The impact of the quick wins on the current projects is listed.

The last columns are listing the responsible body for the roadmap topic, an estimated timeline and an indication if the topic is suitable for separate funding.

The objective of this table is to give insights how current projects could benefit from the developments derived from the technical roadmap. These quick wins should not result in any postponement of current projects.

* = independent project (to be)
** = impact of quick wins
*** = could/should be adopted without delay of the current projects

Technical Services = MSG + ISG + other implementors
Projects = extension project and/or MVD development projects

IFC related activity	Priority	Benefit	Impact** on Software implementation	1. Quick Wins*** for current projects	Impact** on current projects	2. Responsible	3. Final/full IFC Standard*	4. Final/full Other Standard*	5. Suitable for separate funding
IFC Modularisation	high	Easier to maintain; faster release cycles				Technical Services	2 years		
Split base and modules	high	Automatic ('instant') interoperability between modules & domains	Predictable and stable base for a number of years, with dynamic modules that can be supported very quickly.	Faster and easier implementation of multiple extensions with a single code-base	Other way do describe same content (transition will be done by MSG)	Technical Services	6 months		
Create modules	medium/low	Separate release cycles per domain				Projects			yes, in projects
Language independent and consistent IFC	high					Technical Services	1 year		
Language independent	high	Same IFC data independent from the file used file format	Provides wider option of tools and programmers that can be used to implement IFC	Faster and easier implementation of the shared base of IFC	More vendors could be attracted to implement IFC	Technical Services	6 months		
Remove Inconsistencies (general)	medium	More reliable software implementations	Predictability provides savings in implementation recources	Faster and easier implementation of the shared base of IFC	Very close collaboration with vendors. Vendors need to be part of decision making process of updated IFC	Technical Services	2 years		
Removing redundancy (general)	medium	More reliable software implementations	Saving recourses	Vendors will need to do less to implement IFC	Very close collaboration with vendors. Vendors need to be part of decision making process of updated IFC	Technical Services	2 years		
IFC Release strategy	high	Predictable IFC delivery for vendors and users					2 years		maybe
Defining & documenting conformance levels	high	Clear proposition how to use and implement IFC							
One Star	high	Guarenteed interoperability between domains, not matter what module/extension is implemented	One implementation to support multiple use-cases saves recources	Automatic exchange possible (for reference view use-case) between extensions	Harmonisation between Reference Views needs to happen	Technical Services	1 year		no
Two and Three Stars	medium	Additional requirements clearly split with most common use-cases	One implementation to support multiple use-cases saves recources			Technical Services + Rooms & Projects	2 years		
Product Library IFC (parametric geometry)	medium/low	IFC for product libraries				Technical Services	> 2 years		yes
IFC geometry fixes	high	More reliable software implementations	Saving recourses	Vendors will need to do less to implement IFC	Very close collaboration with vendors. Vendors need to be part of decision making process of updated IFC	Technical Services	2 years		yes
Licensing	medium	Clear proposition of possibilities						> 1 year	yes
Setting up new Software Certification structure	high	Faster process; more guarentees for end-users	More predictable outcome			Technical Services		2-3 years	yes
Modelling guidelines 'IFC in UML'	high-depending	Strict way to contribute to IFC development		Already available		Technical Services, based on work in Rooms & Projects		> 1 year	yes
IFC on GitHub	high-depending	Transparant, tracable and version managemed development of IFC	Ability to contribute fixes and changes	Advise is to keep using EA Cloud		Technical Services & bSI MO	6 months		yes
Information Delivery Specification v0.1	high-depending	Reliable definition of exchange requirements	Dynamic configuration of IFC Export; Automatic validation for IFC import			Project (Technical Room)		6 months	yes
Develop CI tools for IFC	high-depending	Quality checking and quality assurance of IFC developments				Technical Services		6 months to 2 years	yes
Creating additional serialisations for IFC	medium	Multiple file formats broaden the adoption of IFC in multiple domains	More options to implement IFC closer to the use-cases			Projects (Technical Room)	depending on projects		yes
CI tools for IFC Quality checking & documentation	medium	Clear overview of quality checking process and workflows, supported by automatic checking tools where possible				Technical Services	1 year		yes
Translations framework for IFC	medium	Dynamic translations that are more up-to-date and easier to fix (by chapters)	Better usability of IFC for end-users			Technical Services (framework) & Chapters (translations)	1 year		yes
Information Delivery Specification standard v1.0	medium-depending	Reliable definition of exchange requirements	Dynamic configuration of IFC Export; Automatic validation for IFC import			Project		> 1 year (CEN Standard 2 years)	yes

A larger version of this overview is available on ShareFile.